Universidade do Minho
Escola de Engenharia

Pavel Vik

Integration of Database, Simulation and CAD towards Assisted Design of Production Systems

December 2011

Integration of Database, Simulation and CAD towards Assisted Design of Production Systems

Pavel Vik

UMinho | 2011

Universidade do Minho
Escola de Engenharia

Pavel Vik

Integration of Database, Simulation and CAD towards Assisted Design of Production Systems

Doctoral thesis of Industrial Engineering and Systems

Work done under the supervision of
Luís Miguel da Silva Dias
and
Guilherme Augusto Borges Pereira

December 2011

Acknowledgements

I wish to express my deepest gratitude to my supervisors, Luís Miguel da Silva Dias and Guilherme Augusto Borges Pereira, for their guidance, patience, constructive criticism and valuable suggestions that I have been constantly receiving from them. The same I wish to express to Assoc. Prof. František Manlig who gave me incredible support during my study period at Technical University of Liberec. It is my pleasure to thank my colleagues and professors at University Minho, as well as my colleagues at Technical University of Liberec for many fruitful discussions and support during the course of this thesis.

I would like to thank all people from companies Blaupunkt Bosch, Cachapuz, Magna, Modus and Škoda, namely Ricardo Araujo, Carmen Belo, Ricardo Abreu, Cândido Martins, Josef Nosek, Vladimír Jareš and Jiří Štoček.

My thanks also go to department secretary Conceição Marques and Acácio Costa from IT help, as well as to faculty administrative staff Daniela Stejskalová and Jaroslava Krejčová for their patience with my administrative messiness.

I would like to thank my friends for their support, to Bahi, Docent, Hofik, Hrstic, Jitka, Kuřátko, Martina, Michaela, Pája, Quentino, Slepic, Václavka, Zinet, Zlát'a and many others.

Last but not least, I want to thank my parents and my brother for their incredible patience and unfailing support during the research.

Pavel Vik
Braga, December 2011

Resumo

O Projecto de Sistemas de Produção é o tema desta tese. Actualmente existe pressão acentuada para as tarefas relacionadas com o Projecto de Sistemas de Produção, no sentido de serem desenvolvidas ou reorganizadas de forma rápida e eficiente. O mercado global e o rápido progresso dos processos produtivos a isso obrigam. Neste contexto tão dinâmico, a flexibilidade, modularidade e robustez são propriedades muito relevantes para os sistemas de produção.

No que diz respeito ao projecto de sistemas de produção, têm sido usadas três categorias de ferramentas computacionais: Projecto Assistido por Computador, Simulação de Processos e Sistemas de Informação. No entanto, estas ferramentas têm sido usadas com baixos níveis de integração.

A ausência de integração de informação entre estas três categorias de ferramentas, e também a ausência de uma abordagem sistémica ao Projecto de Sistemas de Produção têm evidenciado duplicação de trabalho, desperdício de tempo, incoerências, dificuldades de comunicação entre os membros da equipa de projecto e erros na fase de projecto.

Nesta tese, as ferramentas de computação para o projecto de sistemas de produção são integradas numa arquitectura de sistema único. Há uma necessidade de coerência de informação entre as diferentes ferramentas, explorando-se formas de lidar com a diversidade de informação e assegurando sistemas de produção válidos e eficientes, extraindo vantagens da referida integração de informação.

Esta integração foi implementada em MS Access (sistemas de informação e bases de dados), AutoCAD (projecto de layouts) e WITNESS (simulação). MS Access assegura estrutura aberta para a base de dados, permitindo a especificação completa de sistemas com todas as propriedades estruturais e comportamentais e a integração e troca de informação entre WITNESS e AutoCAD. A simulação auxilia o propósito da análise do comportamento dinâmico e estocástico dos sistemas e o CAD auxilia a tomada de decisão de layout em implementações exequíveis.

Iterativamente os resultados das simulações são usados para melhorar o projecto de layout no CAD, e os layouts do CAD são usados para novas experimentações em simulação. Esta abordagem suporta optimização global do sistema de produção, tendo em consideração todos os recursos do sistema, suas restrições e todas as medidas de desempenho do sistema.

Uma ferramenta computacional para este objectivo de integração é então discutida e desenvolvida. Tendo em consideração as referidas desvantagens associadas às conhecidas abordagens ao Projecto de Sistemas de Produção, a ferramenta deverá incorporar um elevado nível de automação. De facto, a ferramenta proposta - *Integrated Design of Systems* (IDS) envolve um amplo conjunto de funções para o desenvolvimento das mais usuais tarefas associadas ao

Projecto de Sistemas de Produção, desde o nível conceptual até ao nível da implementação: análise do sistema de produção (Pareto (P-Q), *Cluster* e análise do fluxo de materiais), geração automática de modelos de simulação, geração de layouts alternativos para instalações e plantas de fábrica, visualização de fluxos de materiais, projecto de sistema de transporte e especificação iterativa de dimensão de *buffers*.

O IDS apresenta diferentes alternativas e providencia informação detalhada sobre medidas de desempenho de sistemas de produção, permitindo a opção pela melhor solução.

O IDS proporciona uma exploração mais sistemática do espaço de soluções e quantifica o desempenho de cada alternativa, potenciando-se a obtenção de melhores soluções que as obtidas com abordagens não-integradas.

O IDS é aberto, permitindo que diferentes organizações possam utilizar esta ferramenta avançada para o Projecto de Sistemas de Produção, explorando as vantagens da simulação e do CAD e da sua integração.

O conceito IDS e as suas funcionalidades foram validadas através da implementação em diferentes projectos, em diversas áreas de aplicação (empresa cimenteira, indústria automóvel, reorganização de chão de fábrica, sistema logístico interno e projecto para Campus Universitário).

Abstract

This thesis deals with production systems design. Nowadays, there is a great pressure on production systems design to be developed or reorganised rapidly and efficiently due to the worldwide competitive market and rapid progress in manufacturing processes. In this dynamic context, flexibility, modularity and robustness are desired production system properties.

As far as Production Systems Design is concerned, three basic classes of software tools have been used: Computer Aided Design, Process Simulation and Information Systems. However, these software tools have been used with low levels of integration. The absence of data integration within these three classes of software tools, and also the absence of a systemic approach to Production Systems Design have been causing duplication of work, waste of time, incoherencies, difficulties in project team communication, and errors in the design phase.

In this thesis, production systems design software tools are integrated into an unified system architecture. There is a need of data coherence between different software tools, exploring ways of dealing with data diversity and assuring valid and efficient production systems, taking advantage of the mentioned data integration.

This integration is implemented on MS Access (information systems and databases), AutoCAD (layout design) and WITNESS (simulation). MS Access provides open database structure, allowing integration and data exchange between WITNESS and AutoCAD. Simulation helps on dynamic systems analysis and CAD on static arrangement on a feasible implementation. Iteratively the results from the simulations are used to improve CAD layout design, and CAD layouts are used in new simulation experiments. This approach supports global system optimization that considers all important system resources and system performance measures.

A software tool for this integration is then discussed and developed. Taking into consideration the referred disadvantages of known approaches to production systems design phase, the software tool should acknowledge a high automation level. In fact the proposed *Integrated Design of Systems* tool (*IDS*) involves a wide set of functions for the most common tasks of production systems design, from conceptual level to implementation level: systems analysis (P-Q, cluster and material flow analysis), automatic generation of simulation models, generation of facility and factory layouts alternatives, material flows displaying, transportation system design and iterative buffer size specification.

IDS shows several alternatives and provides detailed information on production systems performance measures, enabling to choose the best solution.

Solutions achieved using IDS are expected to be better than solutions obtained with non-integrated approaches.

IDS approach is open and accessible, thus enabling different companies to use this advanced production systems design tool, taking advantage of simulation and CAD systems and their integration.

IDS concept and functionalities have been validated through the implementation in several different projects, in different application areas (cement factory, automotive industry, re-layout of job-shop layout, internal logistic system and design of a university campus).

Contents

List of figures

List of tables

List of acronyms

AGSM	Automatic Generated Simulation model
ALDEP	Automated Layout Design Program
BOM	Bill of Material
CAD	Computer Aided Design
CORELAP	Computerised Relationship Layout Planning
CRAFT	Computerized Relative Allocation of Facilities Technique
CT	“Colour-Type”, internal name for product (car bumper)
DB	Database
DCA	Direct Clustering algorithm
DDE	Dynamic Data Exchange
DES	Discrete Event Simulation
DF	Digital Factories
DM	Deterministic Model
ERD	Entity Relationship Diagram
ERP	Enterprise Resource Planning systems
FIFO	First-In-First-Out
FLD	Facility Layout Design
FLP	Facility Layout Planning
HR	“High-runners“, specific marking for large-size production batch
ID	unique number
IDS	Integrated System Design
JIT	Just in Time
LR	“Low-runners“, specific marking for small-size production batch
MR	milk-run
.NET	software framework supports several programming languages
NGFL	Next Generation Factory Layout
OLE	Object Linking and Embedding
PFA	Production Flow Analysis
PSPD	Production System Planning and Design
PQ	Product-Quantity analysis
ROC	Rank Order Clustering algorithm
SLP	Systematic Layout Planning

SQL	Structured Query Language
TCR value	Total closeness value
VBA	Visual Basic for Application
WB	Weight bridge
WCL	WITNESS Command language
WIP	Work-in-Process
WS	Workstation

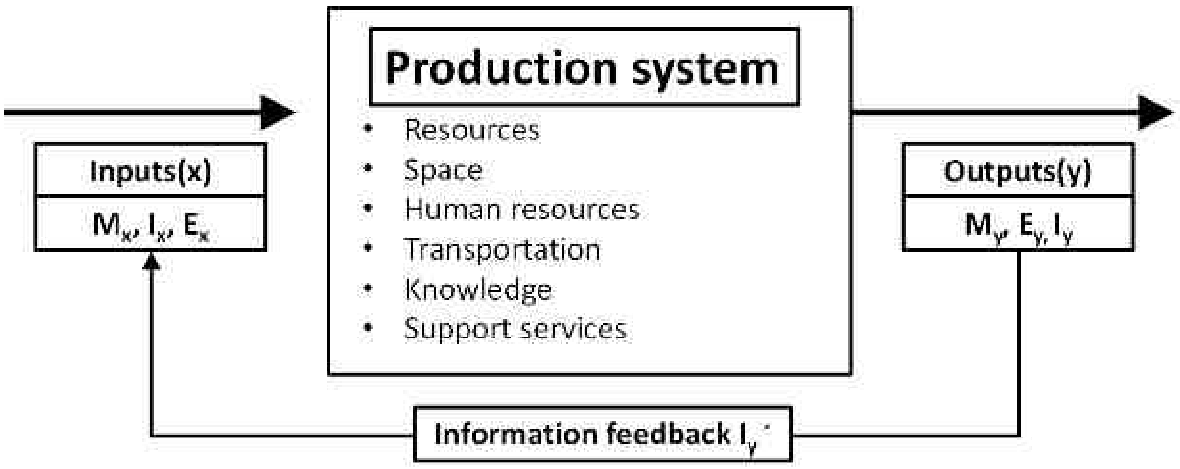
1 Introduction

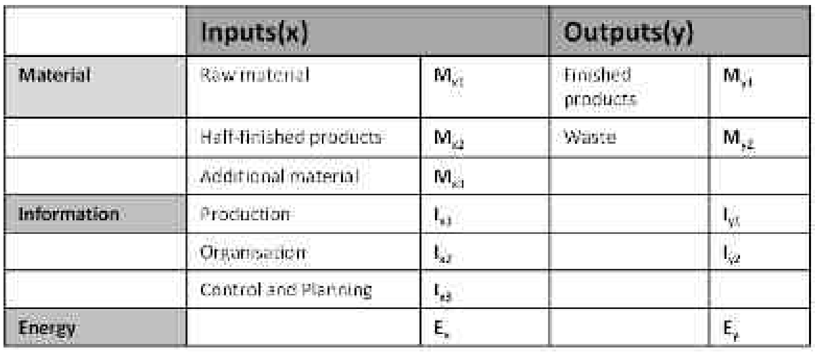
This chapter will describe the problem, reveal motivations and state objectives. The outline of the thesis is showed in the last section.

1.1 Production Systems Concepts

1.1.1 Production systems

A system (production system) is organized as a set of *elements* and *relationships* among them. System *behaviour* is based on the internal organization and is influenced by the external environment. Relationships between system and environment are called “Inputs” (stimulations) and “Outputs” (reactions), as shown in Figure 1 (Zelenka & Král, 1995).





	Inputs(x)		Outputs(y)	
Material	Raw material	M_{x1}	Finished products	M_{y1}
	Half-finished products	M_{x2}	Waste	M_{y2}
	Additional material	M_{x3}		
Information	Production	I_{x1}		I_{y1}
	Organisation	I_{x2}		I_{y2}
	Control and Planning	I_{x3}		
Energy		E_x		E_y

Figure 1 – Schema of the production system

In this context, **Inputs** are material, energy, areas, information, capital, time, etc. All those inputs come from several different sources. Within the production system, Resources can be production machines or other equipment. Space is used for storing material in buffers, storages or warehouses. Human resources are workers that manage all the operations in production, transportation, support services and management supervision. Transportation means the material handling and the manipulation of the operation with forklifts, milk-runs, buses, trucks, etc. Knowledge means all the information about the system, for instance schedules, operation and technological data, controlling programs, facilities location, etc. Support services are all the activities associated with the mentioned elements and activities, that can be engineering processes (product development, testing), facility planning and design, etc. Finally Outputs from the system can be the final product as well as waste, information about products, etc. Other specific outputs can be requested information from database or energy in the power station system (Taylor, 2009).

Figure 2 displays a generic schema of production systems with material (raw material, components, final products and waste) and information flows (production orders) between the most important areas (warehouses, storages, production, assembly etc.)

Production System Planning and **Design** (**PSPD**) is a complex set of tasks using knowledge from several fields: scientific, logical, economical, management, statistical, technical and information technology. It consists of planning and evaluating different **alternatives** of systems aiming the global **optimal usage** of inputs and all kinds of resources. Alternatives are designed regarding dynamic time changes aspects and stochastic influences (Francis & White, 1974)(Heragu, 2006).

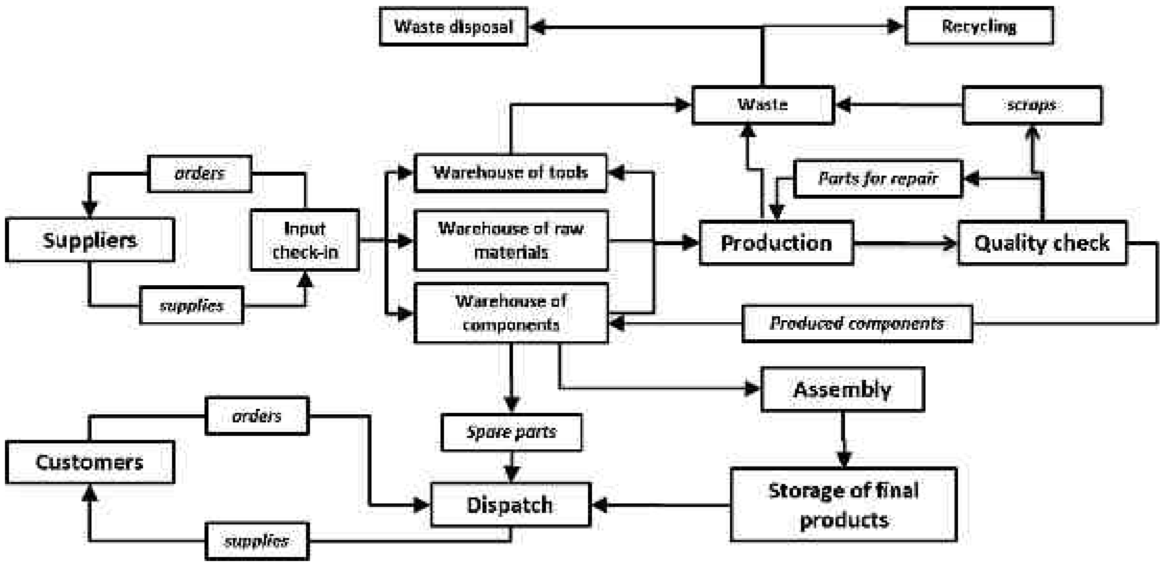


Figure 2 – Production system, adapted from (www.ipaslovakia.sk, 2011)

In the industrial engineering terminology, PSPD is also called "**Facility planning**" (**FP**) or "**Facility Layout Design**" (**FLD**)(Tompkins, White, Bozer, Yavuz, & Tanchoco, 2010).

The terms *Facility Planning* and *Facility Design* have no universal meaning among different authors. In this thesis, Facility Planning is more abstract (conceptual level) and it precedes the Facility Design (implementation level).

„…Facilities planning is the planning and design of the physical environment of an activity to best support the execution of this activity. Facilities Design attempts to organize the tangible fixed assets of an activity in such way as to provide maximal support for the achievement of the activity's objectives at the present time and in the future at the lowest possible cost…„

(Goetschalckx, 1992)

Facility Layout Problem (FLP) can be defined as a way to find optimal arrangement of resources in order to provide an efficient operation (Hassan & Hogg, 1991).

1.1.2 Facility Planning procedure and objectives

The complete Facility Planning procedure contains different levels of hierarchical aggregation, as summarised in Table 1:

I. Factory location
II. Factory layout
III. Facility layout
IV. Design of workstation

Facility planning starts with location phase. It is the process of finding the optimal location in the world: a country, a district, or a space in the current plant, depending on the kind of project and its size. It is influenced by external conditions and factors (e.g. economic – customers, prices, taxes, available space, marketing, macroeconomics, business analysis, etc.)

The next phase is a design of the **block layout**. There are planned activities in the production system, the arrangement of departments in the layout and the planning of their sizes.

The purpose of block layout is to explore several different ways to arrange layout alternatives. This assures that the best overall arrangement is selected.

Detailed layout comes from the selected block layout. The main purpose is to arrange machinery and other equipment inside each area or department.

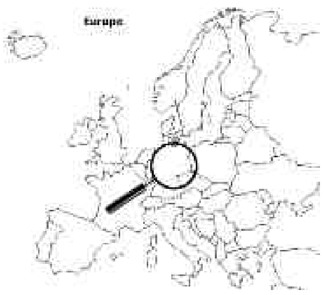
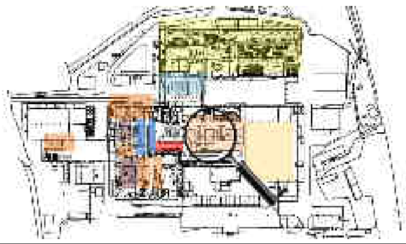
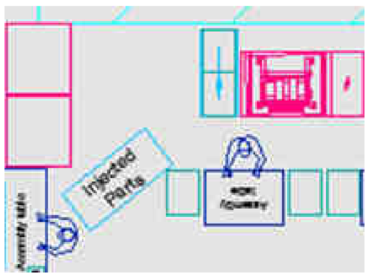
The final phase includes designing some layout details (e.g. ergonomics of workplaces) and **the installation** of the equipment in the plant.

Objectives of Facility Planning and Design

The main objective of the facility planning is to enable the **manufacture of the product**, economically and in the required volume, variety, quality and time. By "economically", it means **reducing material handling** and any kind of wasting, with **effective use** of human power, space and infrastructure.

The design of layouts has been changing for the past sixty years. In the early 50s, the main trends in designing and optimization were based on deterministic material flows with the aim to reduce material flow costs (***Static Facility Layout Problem***).

Table 1 – Levels of layouts design

Level of FP	Description
I. Factory location 	Factors: • Closeness (to the market, to raw materials, to suppliers, to other facilities, to competitors) • Geographical area (transportation access, labour, demographics, climate, environmental considerations) • Economic factors (initial and subsequent costs, salaries, taxes, political support) • Human resources (skills, knowledge)
II. Factory/Plant layout 	Departments allocation: • Relative location and sizes of the planned departments (**block layout**) • Internal logistic system (handling systems design) • Facilities system planning (production and production support areas)
III. Facility layout	Exact location of all equipment and storage areas (**detailed layout**): Layout design (**basic layout patterns**): • Production line layout • Fixed product layout • Cellular layout • Process layout Handling system design
IV. Workstation layout 	Detailed design of each workplace and workstation: • Detailed technical solution • Ergonomics (light, noise, temperature, etc.) • Tools and parts placement to better work performance • Poka-yoke methods to avoid human error

Later, the design and optimizing of layouts considered also re-layout costs; product mix and volumes were divided into several production periods (***Dynamic Facility Layout Problem***).

Nowadays, market trends and customers' habits have been changing considerably, involving now greater variety of products, product customisation, shorter product lifecycles, and more technological difficulties. Also markets have been changing from local to global markets and global concurrency. There is a great pressure on production systems design to be developed or reorganised rapidly and efficiently due to the worldwide competitive market and rapid progress in manufacturing processes.

Due to these factors, different types of layouts are being developed, called ***Next Generation Factory Layout***, NGFL) (Kulturel-Konak, 2007).

Main features of NGFL are mentioned in Table 2. In this dynamic context, flexibility, modularity and robustness are generally highly desired production systems properties. These properties are strongly influenced by uncertainties that can be divided into two groups: internal and external. Internal factors can be **breakdowns** of machinery and equipment, **stochastic** operation and transportation times, waiting times in the queues, routing of parts, **quality** of products (scraps and rework) or human **errors** etc. External factors are related to delays in supplying raw material, uncertainties in customers' **orders**, changes of products' **prices**, macroeconomic factors or current customer trends.

Table 2 – The main desired features for a production system

Feature	Description
High system performance and cost efficient layout	Optimal utilisation of resources (equipment, facilities, services, space) Minimum transportation costs (material handling) Reduced critical system points (bottlenecks) Optimal level of WIP[1] Throughput times
Flexibility	Flexible facilities are able to handle a wide product variety without requirement of re-layout or without affecting significantly the system performance.
Robustness	Ability to accept any kind of product volumes High productivity of large scale production Robust layout could not be optimal for any of the scenarios but it is suitable for all of them
Modularity	Modularity of facilities – division of production systems into segments that support easy facility upgrades, testing, setup, partial breakdowns etc.) Product modularity – different products can be assembled from a small number of modular components
Upgradeability and rapid reconfiguration	Rapid system reconfiguration due to equipment's upgrade or other structural changes
Expansion	Ability of system production expansion with minor impact

[1] WIP (*Work –In – process*) is the total amount of products in the system, either being processed or stored in the buffers.

1.1.3 Factory and Facility layout

Factory and Facility layouts include **physical allocation** of the resources in the available space.

„…Plant Layout embraces the physical arrangement of industrial facilities. This arrangement (installed or planned) includes the space needed for material movement, storage, labourers and all other supporting activities or services, as well as for operating equipment and personnel…„
(Muther, 1973)

„…A facility layout is an arrangement of everything needed for production of goods or delivery of services. A facility is an entity that facilitates the performance of any job. It may be a machine tool, a work centre, a manufacturing cell, a machine shop, a department, a warehouse, etc…„ (Heragu, 2006)

The design of both types of layouts has very similar rules and aims, and both are designed to achieve the desired features mentioned in Table 2. Differences between Factory (Plant) and Facility layout are in the level of design details, where the methods used are similar.

Factory layout is focused on an allocation of factory departments (e.g. production area, warehouse, offices etc.) that are represented by simplified blocks. Therefore, this types of layouts is also called *block layout* or *general layout.*

Facility layout is an allocation of all resources of the given department required for the department's purpose. For instance, facility layout of production department involves the allocation of machines, workstations, buffers, transportation aisles within the department, etc. Resources are represented by a drawing with realistic shapes (Tompkins et al., 2010).

Both layout design phases are dependent on each other and so they must be joined by a **feedback** to affect design changes.

In Figure 3, there is a graph displaying the increasing costs of design changes. Costs for changes in already built production system (e.g. more space for warehouse) proved to be much higher than in designing phases (Marcoux, Riopel, & Langevin, 2005).

"Cost of design changes increase exponentially as project moves beyond the planning and designing phases."
(Tompkins et al., 2010)

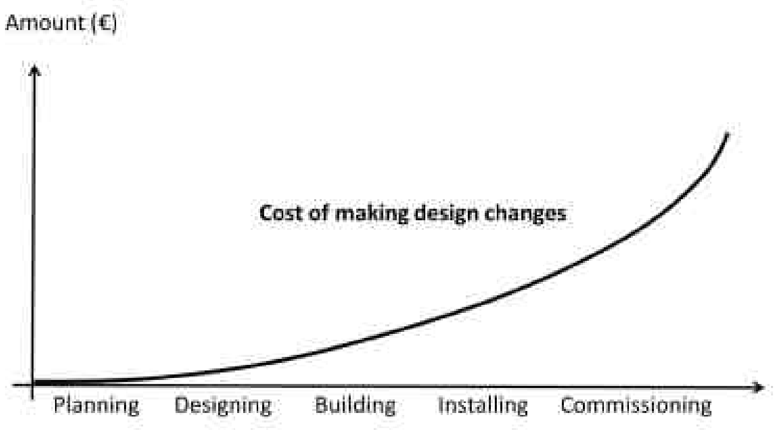


Figure 3 – Cost of making design changes, adapted from (Tompkins et al., 2010)

Layouts are designed for the initial conditions of the business but after changes in markets, technology or production volumes, these changes must be adapted in the initial layouts to achieve a better configuration. This process is called **re-layout**; see more in Section 1.1.4.

In Table 3, there is a list of the most important factors that influence layout and its design. It is necessary to design the production system in order to cover all of them simultaneously.

Table 3 - Factors affecting factory layout

Factors	Description
Products	weight, dimensions, shape, physical-chemical characteristics and other properties
Production	production volumes of each product and product varieties
Manufacturing methods	technology, sequence of the operations
Machinery and production equipment	set of available operations, kind of technology, dimensions, height, quantity, personal requirement (number of workers, ergonomics, safety), requirements of auxiliary services
Material Handling	Transportation of products (receiving, storing, packaging and shipping), personnel transit system, postal system, information system. Influenced by manufacturing method, products, machinery and storage (kind of transportation, used equipment). Transportation batches
Space and storages	Available space for a factory or department, involving all equipment Space for storages and buffers.
Human power	Number of workers, kind of supervising, skill levels, personal and social requirements (rest rooms etc.)

Material Handling is a very important factor that influences the production system and its efficiency. It is a mechanism needed to satisfy the required facility interactions and support production. It is necessary to minimize material handling, eliminating unnecessary and costly movements and use the appropriate kind of equipment for handling.

„…Materials handling equipment and the facilities it operates can contribute to as much as 15- 70 percentage of the total cost of the manufactured product…„

(Tompkins et al., 2010)

„…The best material handling is no handling…„

(Sims, 1990)

Storing of product in any state (raw material, semi-processed or final products) is usually combined with extra costs. In some cases, it is necessary to store products:

- To overcome uncertainties (stochastic operation times, machine breakdowns etc.)
- To enable a safety stock level to avoid material flow stopping
- To synchronise and balance the production system

1.1.4 Facility layout formats

Facility layout formats generally depend on the products variety and the production volumes (Dilworth, 1992). A detailed description is in Section 2.1.1 - P-Q analysis (Pareto analysis).

It is possible to define the following layout formats based on general pattern of work flow and types of organization:

- Process layout (job-shop)
- Product layout (flow-shop)
- Fixed position layout
- Group technology layout

In Table 4, there is a detailed summary of the mentioned layout format and comparison in different aspects such as pattern of demands, product variability, flexibility of production system, material flows and material handling, sizes of buffers or productivity and utilisation of equipment.

Table 4 – Table of layout patterns properties

Kind of layout	Kind of product /demands	Flexibility	Supervisors	Human resource and operators	Production planning and controlling	Material flows and material handling	Buffers	Throughput time	Productivity and utilisation	Tied capital
Process layout	Very small number of product. Single part production	High degree of flexibility. Applicable equipment for a great number of tasks	Highly knowledgeable about their functions	The diversity of tasks offers a more interesting and satisfying occupation. They must have broad skills and must be paid higher wages than assembly line workers	Very complicated job shop control	backtracking long movements	Necessary to avoid workers waiting for jobs	Higher (waiting, complicated material handling)	Lower, because of equipment (machinery) set-up and operator training before each job. Lack of efficiency in timing as workers must wait between tasks	Low prices of fixed equipment (lower investments). Large amount of WIP. More space for buffers, transportation
Product layout	Standardized product. Large stable demand. Minimum improvements or changes	Lack of process flexibility, since the layout is determined by the product; a change in product design may require major alterations in the layout. Lack of flexibility, as the product cannot flow faster through the line until the slowest task is accomplished (unless that task is performed at several stations)	Supervision is general, rather than specialized	Little skill is usually required by operators of the production lines, hence training is simple, short and inexpensive. Worker fatigue as workers may become bored by the endless repetition of simple tasks	Simple production planning control systems are possible. Flow ... line balancing exercise	Smooth and logical flow lines because equipment arrangement corresponds to the sequence of operations. Since the machines are located so as to minimize distances between consecutive operations, material handling is reduced	Almost no buffers	Total production time per unit is short.	A breakdown of one machine or the absence of enough operators to start all work stations may stop the entire line.	Since the work from one process is fed directly into the next, small in-process inventories result. Comparatively high investment is required as identical machines (a few not fully utilized) are sometimes distributed along the line.
Cellular Layout	Product can be grouped into families of similar parts	Reduced shop flexibility	General supervision required	Increase operator expertise. Higher skills level required of employees than for product layout		Reduced material handling	Depends on balanced material flow between product layout and process layout; otherwise buffers and work in process storage are required		Reduced tooling. Reduced setup time. Lower machine utilization than for process layout. Enhanced job satisfaction	Support the use of general purpose equipment
Fixed Position Layout	Physically large and heavy products. Low sporadic demand	Highly flexible; can accommodate changes in product design, product mix and product volume	General supervision required	Promotes job enlargement by allowing individuals or teams to perform the whole job. Higher skill requirements for personnel as they are involved in more operations.	Continuity of operations and responsibility results of a team. This reduces the problems of re-planning and instructing people each time a new type of activity is to begin.	Material movement is reduced, damage and cost of moving are also minimized. Increased movement of personnel and equipment may be expensive. The necessary combination of skills may be difficult to find and high pay levels may be necessary. Cumbersome and costly positioning of material and machinery.			Independence of production centres allows scheduling to achieve minimum total production time. Low equipment utilization as equipment may be left at a location where it will be needed again in a few days rather than moving to another location where it would be productive	

a) Process Layout

In the **process layout** identical workstations are combined in the departments. It is shown in Figure 4 where each block symbolises a specific production machine ("L" as lathe, "M" as mill, "A" as assembly, etc.). This type of layout is very flexible because of the use of general production equipment. The major disadvantages are complex operation control; material handling; the need of high stuff knowledge; and complex material flows (long distances and crossing).

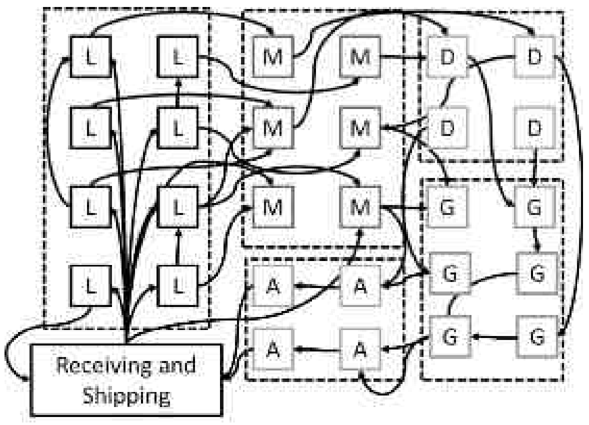


Figure 4 – Process layout

b) <u>Product Layout</u>

In this type of layout (see Figure 5), resources and equipment are arranged into **production lines** to achieve minimum material movement and minimum operation time for the given product with high volume as a consequence of large and stable customer demands. Material flows are fixed and resources are placed in the correct operation sequence. It is the reason why this layout is also called flow-shop. Production resources are often very specialised for given operations with a high level of automation, e.g. using a single purpose machine with specific tools.

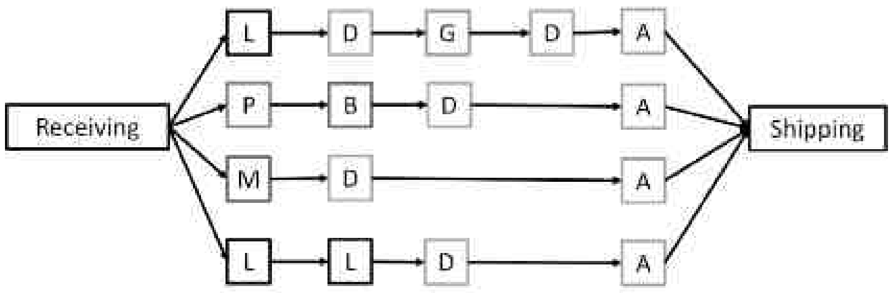


Figure 5 – Product layout

c) <u>Fixed Position Layout</u>

Manufacturing physically large and heavy products (locomotive, airplanes, and ships), with very occasional demands, needs a special type of layout. It is required that the operation must come to the product, rather than the product moving to the machines. So in the **fixed position layout** (see Figure 6), material, workers and equipment are moved as needed, while the product is in a static position.

d) <u>Cellular Layout / Group Technology Layout</u>

Cellular Layout (Production cell or Group Technology Layout) is a type of layout in which machines are grouped to produce a set of similar items or part families that require similar processing (operations), tools, jig or fixtures (see Figure 7). This type lies between mass production and part production. Cluster analysis is an assistant for the identification of part families (Kusiak & Chow, 1987). The description is in Section 2.1.2.

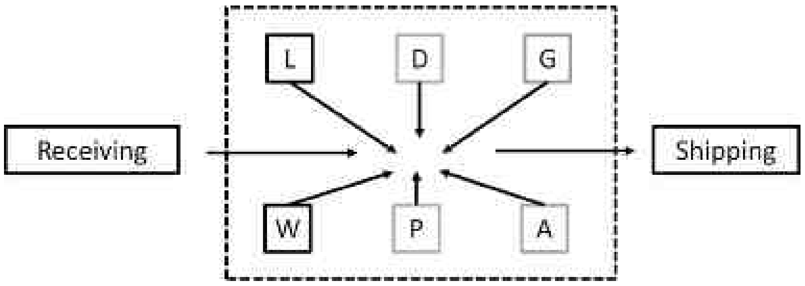


Figure 6 – Fixed position layout

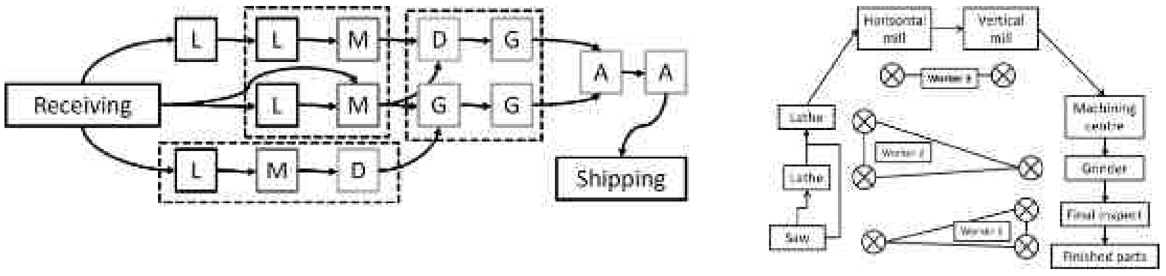


Figure 7 – Cellular Layout

After the four mentioned basic layout patterns, the following non-production type of layouts are also used:

- **Warehouse** layouts – warehouse operations (receiving, storing, order picking, shipping),
- Service layouts - the aim is to minimize flow of information, paperwork and people involved in the services operations (hospitals, banks, libraries, etc.)
- **Supermarket** and retail layouts - maximum items exhibition and stopping time of customers
- **Office** layouts - open office concept (flexible for changes, easier supervision, improved communications)

Auxiliary facility layout areas:

- locations of all machines, employee workstations and other production equipment
- service areas
- material storage area
- aisle, transportation and manipulation areas

- restrooms, lunchrooms, sanitation systems
- internal walls and other building requirements
- offices (administration, supervisors, etc.) and computer rooms
- electrical, water, atmospheric systems and lightning systems
- communication system
- quality testing and developing areas

Figure 8 shows an example of a real production system – workstations containing mouldering machine, assembly processes and storages of inputting and final products. This example demonstrates the space requirements that must be considered for the total required space. The total required area is a sum of machinery, personnel and material space.

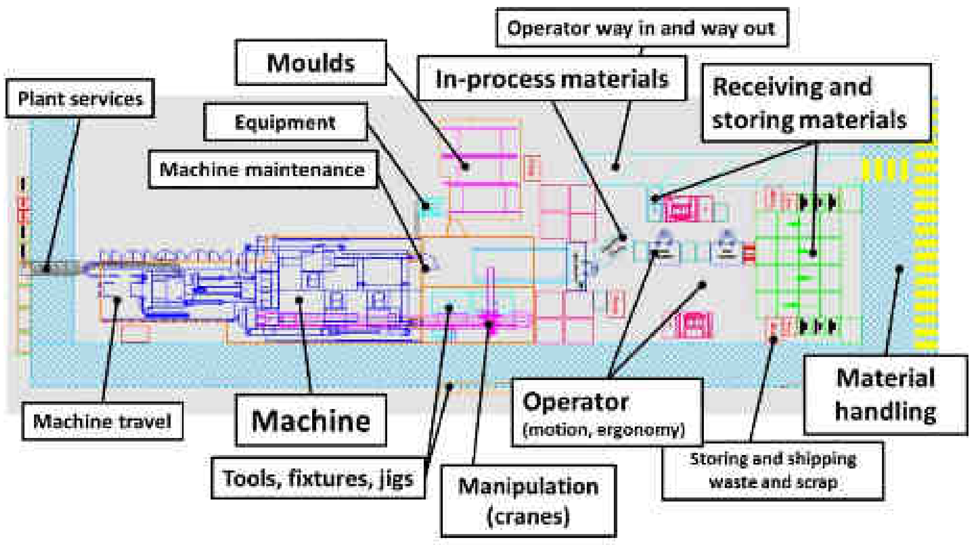


Figure 8 - Facility layout of real production system

1.1.5 Re-layout

Re-layout (or re-design) is the process of changes in the original facility design. It is a very common task of design of layouts. Errors in the initial layout design, inserting new facilities into the current system or influences of uncertainties (mainly external - new jobs, new products etc.) could be some of the reasons for the need of re-layout.

The following symptoms detect a need for re-layout:

- flows and traffic (material congestions, long distances in the work flow process, not straight, crossing, etc.)
- bottlenecks, idle times on workstations
- bad use of space, excessive stocks

- workers (ergonomic problems and bad work conditions, even accidents)
- difficulties in controlling operations and personnel

Facility layouts should be initially designed considering the desired system features (mentioned in Table 2). Flexible, modular and easy re-configurable system couldn't be changed with each new production requirements.

Re-layout can improve system performance but it also involves **extra costs** that can be just temporary, turning the new facility obsolete in short time. In some cases, re-design causes a limitation of the current production system or even **shut down**. These two factors often make plant engineers prefer holding the current inefficient system without changes.

„…In our work with over 20 companies in the last five years, we have encountered mounting frustration with existing layout choices, particularly in companies that offer many products with variable demand and short life cycles. These companies value layouts that retain their usefulness over many product mixes or can easily be reconfigured. Equally important are layouts that permit shorter lead times, lower inventories, and a greater degree of product customization…„

(Saif Benjaafar, Heragu, & Irani, 2002)

1.2 Objectives

This thesis deals with production systems design and its improvement. It is focused on the design of systems and layouts based on material flows, on re-layout processes and also on the design of layouts influenced by different types of uncertainties.

Production systems must be developed or reorganised rapidly, efficiently and with optimal configuration due to the associated worldwide competitive market and rapid progress in many application areas like building industry, manufacturing, information, communication and all types of new technologies. Time between customers' demands and production feedback must be as short as possible. Organisation of the system should provide high performance and cost efficient layouts, such as optimum utilisation of resources, minimum transportation and logistics costs, maximum throughtput rate and minimum delivery times.

Due to high complexity of production systems design, some specific software tools are frequently used; especially three basic classes of software tools are used: Computer Aided Design, Process Simulation and Information Systems. These tools help on improving the design process.

However, these software tools have been used with low levels of integration. The absence of data integration within these three classes of software tools, and also the absence of a systemic approach to Production Systems Design have been causing duplication of work, waste of time, incoherencies, difficulties in project team communication, and errors in the design phase. Thus, it lowers the expected quality of the solution in the available/suitable time to design.

The aim is to conceptualize an integrated approach for system design and develop the correspondent application prototype to validate the concept. In this concept, software tools are integrated into a unified system architecture solution with data coherence between those different tools and taking advantages of the adequate use of the three mentioned classes of tools. This integration (*Integrated Design of Systems, IDS*) will be implemented on a Database system, CAD system for layout design and a simulation tool. Simulation helps on dynamic systems analysis and CAD on static arrangement on a feasible implementation.

Database system should provide an open structure, allowing integration and data exchange between the simulation tool and the CAD system. Database will hold all production system description, including all relevant data from different applications. Database should be independent on the used integrated applications and also its structure should provide designing in different levels of details.

This solution should involve feedback between integrated tools. Thus, changes in the production system configuration managed in one of the applications would take effect in the other application. This means that results from the simulations are used to improve CAD layout designs, and CAD layouts data are inputted in new simulation experiments. This process should work iteratively so as to take advantage of the integrated approach.

Due to mentioned desired IDS features, the software tools should involve a high automation level for production systems design phases. Thus, the proposed solution will involve a wide set of developed functions for the most common production systems design tasks, from conceptual level to implementation level.

IDS will support **automatic generation** of facility and factory layouts alternatives and automatic material flows displaying. Simulation models would then be automatically generated and configured, and simulation results loaded to the database. Simulation model will provide full automatic background simulation. There is also a set of functions for an iterative transportation system design (forklift, milk-run) and an iterative buffer size specification.

These functions are all managed through user forms for input/output data control and results displaying.

This approach supports global system optimization that considers all important system resources and system performance measures.

The overall concept will be implemented on several case studies. It is important to validate this concept on practical examples from different application areas. These projects were chosen so as to test the whole lot of IDS functionalities and principles. It will also be possible to compare traditional approaches with the proposed IDS integrated system.

Thus, the application areas chosen would cover a wide range of real problems: would involve different detail levels for input data – including both factory and facility system design; would also be focused on planning and optimization of material flows; would also include transportation system design for supplying production facilities; would also involve re-layout procedures.

1.3 Outline of the thesis

This Chapter has been dedicated to the description of the problem, with the necessary context and the motivation as a consequence of the relevant challenges associated with the development of an innovative production systems design software tool.

Thus, the following chapter will be dedicated to an overview of the most relevant issues on production systems planning and design. These issues include tools for production systems performance analysis and algorithms for improvement and layout optimization. This chapter will also describe the state-of-the-art of production systems design software tools.

After a thorough review on traditional software tools, emphasizing the lack of integration concepts and consequences on the production systems design process, Chapter 3 will focus on the challenges of the integration concepts anticipated for the development of the IDS system. An initial discussion on possible advantages over traditional approaches is presented.

Chapter 4 is then dedicated to the development of the integration tool. The integration concepts are described and the specific software tools are selected. For a full comprehension of the integration tool developed, this Chapter also illustrates the actual use of specific IDS functionalities involved in input/output user control forms.

The purpose of Chapter 5 is to validate the utilization of the IDS system developed through real case studies of different application areas where different issues of the production systems design phase are tested.

Finally Chapter 6 drives the main conclusions, presents the most relevant contributions of this work and discusses future issues of production systems design.

2 Production systems planning and design

This chapter introduces **Production Systems Planning** and **Design** (**PSPD**) and related tools and methods used for the analysis of a production system, including: *PQ* or *Cluster analysis*, *Relationship* and *From-To charts*, and dynamic methods as *Discrete Event Simulation*. These tools are fundamental to **Facility Layout Design** (**FLD**) and to solving **Facility Layout Problems** (**FLP**), supporting the design in two levels: the *general factory layout* and the detailed *facility layout*. Several procedures have been developed in FLP, such as **algorithmic** (*CRAFT*, *CORELAP*, *ALDE*P etc.) and **procedural** methods (*Systematic Layout Planning*).

In the industrial context, several commercial software tools are used to help industrial engineers with system design processes. These software tools could be divided into several coherent groups based on the provided functions. The most important groups of tools are: **Database** (DB), **Discrete Event Simulation** (DES) and **Computer Aided Design** (CAD). For each type of tool, a set of typical tasks and functions are presented and discussed.

These tools and applications are usually used in projects with different levels of integration: completely separate and independent; with low level of integration; and completely integrated systems (**Digital Factories**, DF).

In this chapter, there is also a summary of the state-of-the-art on integration approaches to production systems design.

2.1 Production systems analysis – an overview

In general, production systems analysis includes *relationship diagrams*, *product-volume analysis*, *production lines analysis*, *process flow charts*, *From-To charts*, *material flow diagrams*, *product routings* etc. These different types of systems analyses help to understand the behaviour of the production system (Francis & White, 1974).

In some cases, material flows or other diagrams have systems that are very difficult to understand because they can contain a lot of information. Therefore, it is better to see the same problem from different angles – for these cases, there are charts that can be helpful, such as *P-Q chart*, *D-I chart* or *From-To table*, which express the same information in a more visual and summarised support (Sly, 1995).

2.1.1 P-Q analysis (Pareto analysis)

One of the most significant production systems analysis is *P-Q analysis* or *Pareto analysis*. "**P**" means product variety and "**Q**" means quantity (the volume of each product). In Figure 9, there is an example of this analysis as a graphic representation (Tompkins et al., 2010) (Heragu, 2006).

Products (in the horizontal axe) are sorted into decreasing order by their production volumes (vertical axe).

It is possible to reclassify products into families with similar properties and P-Q analysis gives basic information about production type and its organisation, material handling, and production planning and control. For many production systems, the "*Pareto law*" states that around 20% of products make around 80% of production volume. If it is not possible to identify this division, then multiple Pareto division can be used (Sly, 1997).

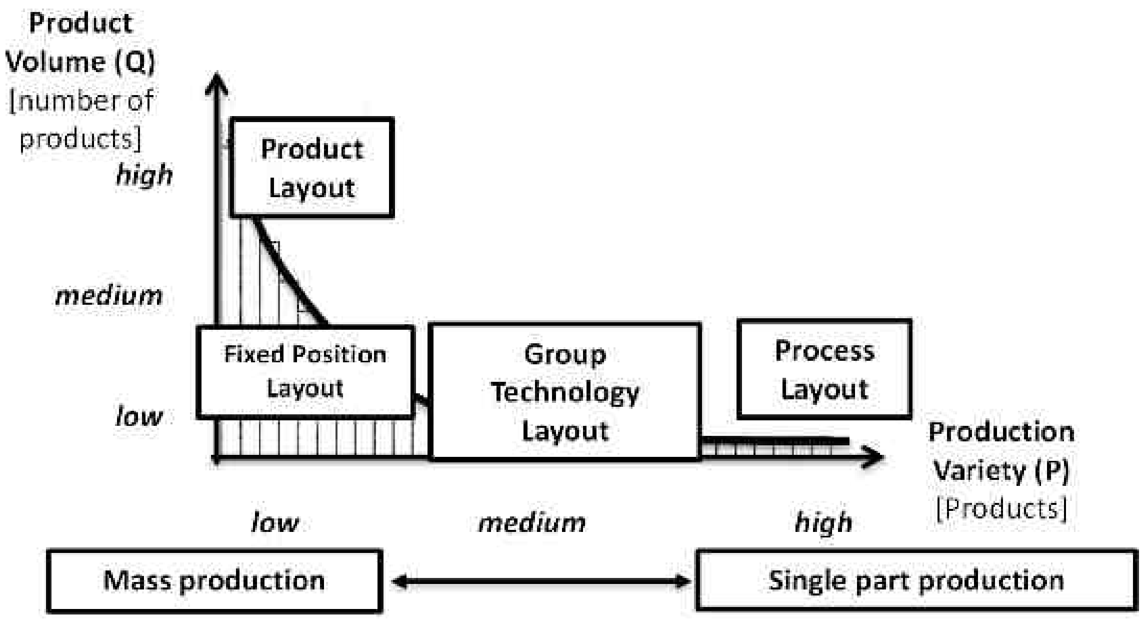


Figure 9 – PQ analysis

Information based on the shape of the curve:

- deep shape – it can be convenient to divide a system into two or more sections (mass and individual production)
- soft shape – one layout for the whole production

With this type of analysis, it is possible to define optimal layout patterns, such as product, process layout, etc. The detailed description of the layout formats is in Section 1.1.4).

2.1.2 Cluster analysis

Cluster analysis represents a set of algorithms for the segmentation of the production. It is mainly used in **Group technology** design (GT, or *Cellular manufacture*, *Part family manufacture*) to aggregate medium volume-variety parts into common families based on similar manufacturing operations, product design properties, etc. (Irani, 1999)(Wemmerliiv & Heyer, 1989)(Sekine, 1992).

Therefore, through GT, it is possible to manufacture similar parts (part families) in different machine groups called cells. Machine groups consist of various technologies and they are almost independent production units (see Figure 7). Benefits from the GT use are very clear – simplified planning of material flows and schedules, reducing WIP in the buffers, reducing material handling, and improving confidence of labours, through a larger variety of tasks and responsibilities, etc. (Baker & Maropoulos, 1997)(Kwok, 1992).

Three basic methods are used for solving GT:

- Visual method
- Coding and classification method
- Production flow analysis method

Visual methods are based on grouping parts based on similar design features. The process of grouping (*classification*) is usually done through comparison of parts based on the user's vision. This method is strongly influenced by the experience of different users.

Coding and classification method is based on geometric shape (dimension, material, shape, etc.). Each feature is represented by a digit code, whereas a full part description is a set of these codes. Codes can be built from a hierarchical structure (*mono-codes*), chain-type structure (*polycodes*) or hybrid coding. The most used systems are *Opitz classification* system (developed by the University of Aachen), *MultiClass* (Organization for Industrial Research) or *CODE* (Manufacturing Data System).

The most common way to identify cell configurations is to use **Production Flow Analysis** methods. These methods are used in cases of layout changes, from process to GT layout. It tries to design independent groups of machines producing only specific parts. C*luster algorithms* as a basic function of *data-mining* (which is a process of discovering hidden information from database) are then used.

Machine component group analysis

In **Production Flow Analysis** (PFA), cells and part families are determined by grouping similar production routings based on PFA matrix (Burbidge, 1971). In this matrix, each row represents a process plan (machines) and each column represents components (parts).

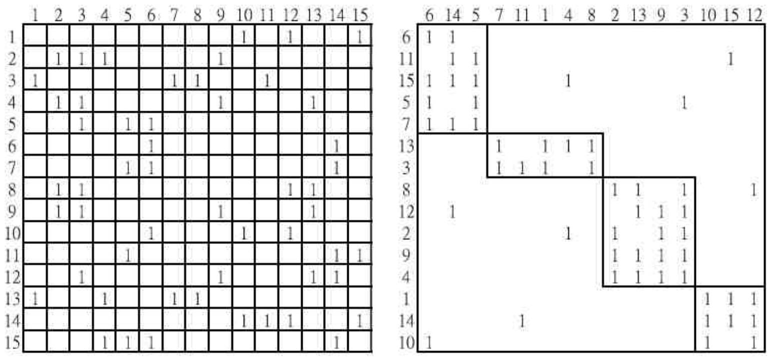
For design cells, *clustering algorithms* can be used, such as:

- Rank Order Clustering algorithm (**ROC**) (King, 1980)
- Direct Clustering Algorithm (**DCA**) (Chan & Milner, 1982)

Similarities in production can be found by *Similarity coefficient methods:*

- **SLCA** (Single-Linkage-Clustering Algorithm or **ALCA** (Average Linkage Clustering Algorithm) (McAuley, 1972)
- **CI** (Cluster identification algorithm) (Kusiak & Chow, 1987)

These algorithms, such as *"The Black Box"* (Baker & Maropoulos, 1997), Group Technology Cell Formation (Yang & Yang, 2008) or even genetic algorithms (Car & Mikac, 2006) (McCormick, Scmwitzer, & White, 1972), use a binary Machine-Part matrix.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1										1		1			1
2		1	1	1					1						
3	1						1	1			1				
4		1	1						1				1		
5			1		1	1									
6						1								1	
7					1	1								1	
8		1	1									1	1		
9		1	1						1				1		
10						1				1		1			
11					1									1	1
12			1						1				1	1	
13	1			1			1	1							
14										1	1	1			1
15				1	1	1								1	

	6	14	5	7	11	1	4	8	2	13	9	3	10	15	12
6	1	1													
11		1	1											1	
15	1	1	1				1								
5	1		1									1			
7	1	1	1												
13				1		1	1	1							
3				1	1	1		1							
8									1	1		1			1
12		1								1	1	1			
2							1		1		1	1			
9									1	1	1	1			
4									1	1	1	1			
1													1	1	1
14					1								1	1	1
10	1												1		1

Figure 10 – Example of a cluster analysis

In Figure 10, there is an example of a cluster analysis where three cells were created from machines (columns) and parts (rows), e.g. machines 6, 14 and 5 produce parts 6, 11, 15, 5 and 7. Some elements don´t fit in only one cluster. Solution for these exceptions, which are not joined to any cluster, could involve duplicating the machines, changing the process plans or even subcontracting these operations.

As an example of the use of these clustering algorithms, a heuristic for ROC algorithm is presented in Figure 11. The input is a binary matrix (machine/part). Binary values $\mathbf{W}_i$ for each row are calculated based on a column index. Rows of binary matrix are rearranged based on values in $\mathbf{W}_i$ in descending order. The same procedure is processed for columns. It is repeated until no changes are possible.

2.1.1 Process analysis

Process analysis also belongs to the basic type of analysis giving important information about the system. It can help to improve current processes by the identification of bottlenecks, wasting processes as repetitive operations, temporary storing, etc. It helps to reduce total throughput time.

Process analysis charts are used for this purpose, where each process has a specific symbol (see Table 5). In Figure 12, some examples of Process Analysis charts are shown.

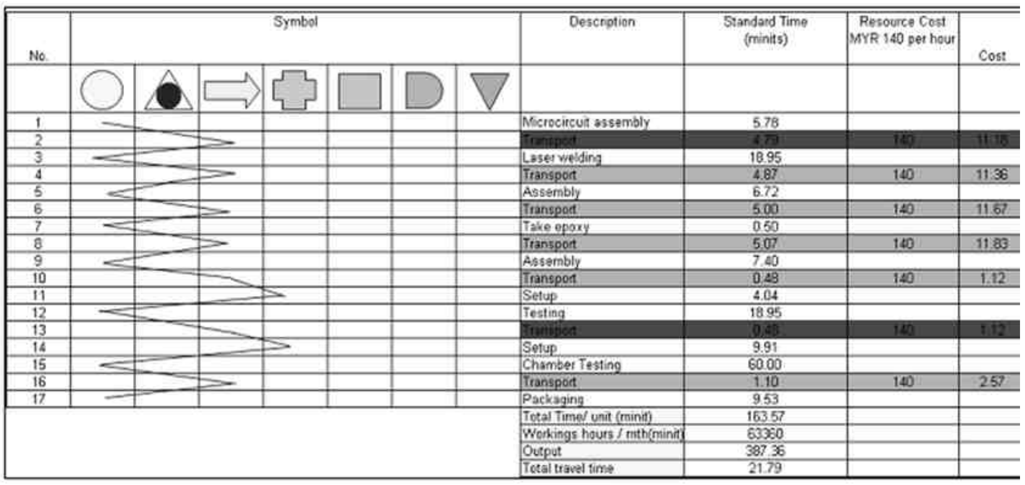
m: total number of columns
n: total number of rows
i; **j**: index of row and column
$\mathbf{b_{ip}}$; $\mathbf{b_{jp}}$: binary value of the matrix (0 or 1)
$\mathbf{W_i}$; $\mathbf{W_j}$: evaluation of each row, column

```
**Initial binary matrix (machine/part)**
**Repeat**
        For each row in matrix:
                $W_i = \sum_{p=1}^{m} b_{ip} 2^{m-p}$
        *Rearrange* rows to make $W_i$ fall in descending order

        For each column in matrix based on:
                $W_j = \sum_{p=1}^{n} b_{jp} 2^{n-p}$
        *Rearrange* columns to make $W_j$ fall in descending order

**Until** no changes in position of each element in each row and column

**Sorted binary matrix**
```

Figure 11 – Heuristics of ROC algorithm

Table 5 – Symbols used in Process analysis charts

Process	**Symbol**	**Description**
Operation	●	*work activity, work on part*
Inspection	▬	*an inspection of the product for quantity or quality control*
Transportation	➡	*movement of material*
Storage	▲	*permanent storage of materials in warehouse*
Delay	◗	*delay in the sequence of operations (waiting for the next operation), temporary storing*



No.	Description	Standard Time (minits)	Resource Cost MYR 140 per hour	Cost
1	Microcircuit assembly	5.78		
2	Transport	4.79	140	11.18
3	Laser welding	18.95		
4	Transport	4.87	140	11.36
5	Assembly	6.72		
6	Transport	5.00	140	11.67
7	Take epoxy	0.50		
8	Transport	5.07	140	11.83
9	Assembly	7.40		
10	Transport	0.48	140	1.12
11	Setup	4.04		
12	Testing	18.95		
13	Transport	0.48	140	1.12
14	Setup	9.91		
15	Chamber Testing	60.00		
16	Transport	1.10	140	2.57
17	Packaging	9.53		
	Total Time/ unit (minit)	163.57		
	Workings hours / mth(minit)	63360		
	Output	387.36		
	Total travel time	21.79		

Figure 12 –Process analysis chart (adapted from (www.ciise.concordia.ca, 2011))

2.1.2 Flow analysis and *From-To* chart

Material and information flows represent movements of products, transportation equipment (e.g. forklifts), human power, information (e.g. kanban cards) and other materials (e.g. tools, waste material) through defined processes in the system (external suppliers, factory, department, and production cell).

Material flows are often combined with internal and external logistic systems, transportation, supplying, etc.

The design of material flows is very important for the overall production efficiency. In some types of production systems, costs involving the logistic system, including transportation costs, could reach 70% of total product costs (Tompkins et al., 2010).

Due to these reasons, flows (transportation distances) must be as short as possible, straight, and with minimum backtracking and mutual crossings. The design of flows is very important for production environments where material transportation is fundamental – heavy material, large-sized objects in large quantities, or high transportation costs (Sly, 1997).

From-To Chart

This square matrix is useful to the analysis of material flows between given departments. It is used as basic input data set for many optimization algorithms focused on minimizing transportation costs (see Section 2.2.1 - Algorithmic methods overview, CRAFT, Graph theory Block Layouts or Block Plan). Typical units of material flows are:

- total number of parts transported by time unit
- weight material flows (number of parts multiplied by their weight per time units)
- cost material flows (number of parts multiplied by cost for transportation per time units)
- number of transported units per time unit (intensity of traffic, number of the trips),

Figure 13 shows an example of ***From-To*** and ***From-Between*** charts. Departments in rows are starting departments ("from") and departments in columns are target departments ("to"). In the ***From-To*** type of chart, direction of flows is known. Using this **From-To** Chart, a complete flow pattern of the system is described. In a **From-Between** Chart, there is only the value for each flow between pairs of departments - without considering direction of flows.

Departments [flows]	Department A	Department B	Department C	Department D	Department E	Department F	Department G	Department H	Department I
Department A	0	90	0	140	0	10	0	0	0
Department B	0	0	90	0	0	0	0	0	0
Department C	40	30	0	180	100	0	40	0	0
Department D	70	0	60	0	210	10	30	0	0
Department E	0	0	0	30	0	90	80	180	0
Department F	10	0	0	10	40	0	0	110	0
Department G	0	0	0	60	0	0	0	0	0
Department H	0	0	40	0	200	70	0	0	40
Department I	0	0	0	0	0	0	0	20	0

Departments [flows]	Department A	Department B	Department C	Department D	Department E	Department F	Department G	Department H	Department I
Department A	0	90	40	210	0	20	0	0	0
Department B	0	0	120	0	0	0	0	0	0
Department C	0	0	0	240	100	0	40	40	0
Department D	0	0	0	0	240	20	90	0	0
Department E	0	0	0	0	0	130	80	380	0
Department F	0	0	0	0	0	0	0	180	0
Department G	0	0	0	0	0	0	0	0	0
Department H	0	0	0	0	0	0	0	0	60
Department I	0	0	0	0	0	0	0	0	0

Figure 13 – *From-To* Chart and *From-Between* Chart

Flows can be drawn in the project layout to improve its visualisation. Several kinds of flows can be displayed, such as:

- **direct flows diagram** (so called Spaghetti diagram) connecting department centres (or input/output places) by direct, unconstrained flows, displaying schematic process flows (see example in Figure 40)
- **aisle flows diagram** displaying flows of material based on real shaped factory aisles (roads, paths etc.). These flows display the actual travel distances. It is useful for the optimization of the equipment location, improving facility layouts, etc.
- **summarised flows diagram** showing additional flows between defined aisles sections. It is helpful in designing the material handling system for the recognition of traffic jams due to the high frequency transportation (see example in Figure 37)

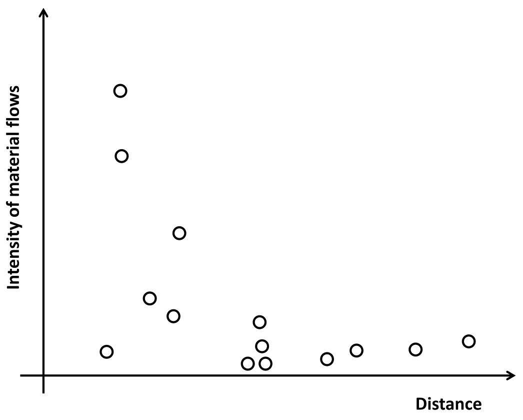


Figure 14 - Distance Intensity Chart (adapted from Sly (1995))

2.1.3 Distance Intensity Chart

Distance Intensity chart (DI) is used for measuring the efficiency of material handling. DI chart of an optimal layout corresponds to points close to the axis while other would mean inefficiency of material flows or handling system. An example of DI chart is presented in Figure

14, adapted from Sly (1995).

2.1.1 Relationship chart and diagrams

Relationship chart

This type of chart is a very useful and effective tool for displaying relationships between each object in any layout (departments, facilities etc.) and it rates the importance of the closeness between them. Results of this chart are displayed in a relationship **diagram** (below). An example of the chart is shown in Figure 15, (www.ciise.concordia.ca, 2011).

Relationships could be:

- **quantitative** (flows, sequences of operations etc.), flow can include material, personnel, information, organization, control etc.)
- **qualitative** (security reasons, supporting services, sharing tools, contamination, supervision, ergonomics – noise, vibrations; dustiness, quality influences, multi-machinery tendering, psychological and sociological influences, etc.)

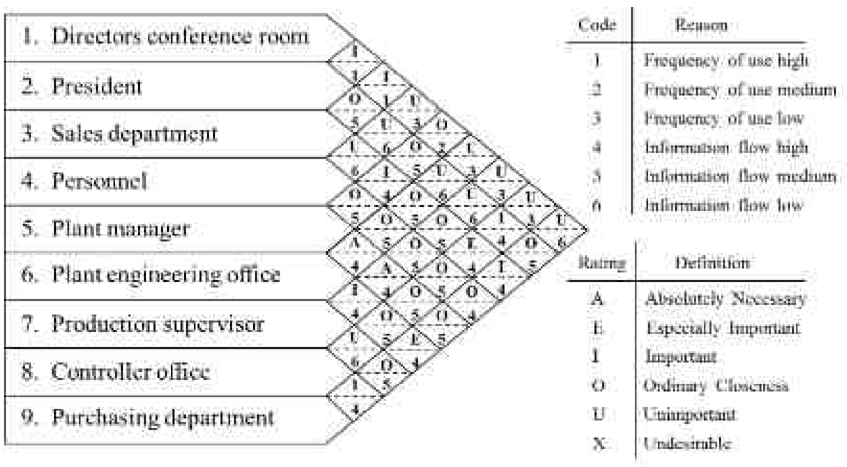
The relationships and the importance of closeness of objects are evaluated by the scale AEIOUX (closeness rating system) where:

- A - **A**bsolutely necessary that the activities be next to each other.
- E - **E**specially necessary
- I - **I**mportant
- O - **O**rdinary closeness
- U - **U**nimportant
- X - activities should not be close to each other (e.g. power hammer and brushing centre)

For the construction of layouts through the use of an algorithm, these alphabet rating system is replaced by numbers and each letter is associated with a specific value – for example: A=125, E=25, I=5, O=1, U=0 and X=-125.

Relationship diagram

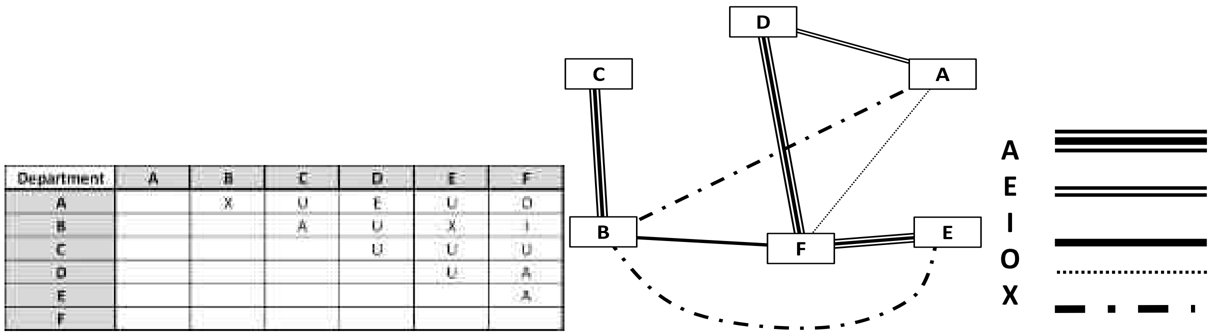
The relationship diagram is based on the data from the relationship chart table. It displays departments (resources) in the available space as simple blocks. Departments with strong interactions are placed close to each other, in adjacent physical locations. In this schematic layout, neither space requirements, nor geometrical shapes nor other constraints are considered. Example of this diagram is shown in Figure 16, where the connection between objects corresponds to relationships.



Code	Reason
1	Frequency of use high
2	Frequency of use medium
3	Frequency of use low
4	Information flow high
5	Information flow medium
6	Information flow low

Rating	Definition
A	Absolutely Necessary
E	Especially Important
I	Important
O	Ordinary Closeness
U	Unimportant
X	Undesirable

Figure 15 - Table of relationships, adapted from (www.ciise.concordia.ca, 2011)



Department	A	B	C	D	E	F
A		X	U	E	U	O
B			A	U	X	I
C				U	U	U
D					U	A
E						A
F						

Figure 16 - Relationship diagram

2.1.2 Dynamic methods

The previously presented methods are based on deterministic type of data. There are also methods using stochastic data, uncertainties in processes and events in specific times. With the implementation of these factors to the dynamic models, it is possible to get a higher accuracy in the results obtained when compared with deterministic models. Also they are suitable for solving more complex tasks – such as controlling logic for material flows.

Using dynamic models to represent real systems, it is possible to analyse system over time and interactions between its elements. It is also possible to get information as throughput, resource utilisation, buffer behaviour, etc.

Table 6 contains some methods and corresponding application area. The most significant analytic method for production systems design and operation analysis is **Discrete Event Simulation** (DES). It is helpful for many different tasks (see more information in Section 2.3.2 - Discrete Event Simulation tools, where simulation concepts and software tools available are described in detail). Simulation models can deal with features of real life problems that are very difficult to consider in static models (Tam & Li, 1991)(Tang & Abdel-Malek, 1996)(Pandey, Janewithayapun, & Hasin, 2000)(Castillo & Peters, 2002) (Keller, 1996).

Table 6 – Dynamic methods

Methods / Analysis	**Math programming**	**Network analyse**	**Queuing theory**	**Theory of storage systems**	**DES**
Reference	(Sniedovich, 2009)	(Kelley, 1961)	(Kleinrock, 1976)	(Gani, 1957)	Description in Section 2.3.2
Keywords of method	method for solving complex problems by breaking them down into simpler sub-problems	applied graph theory , Critical Path Analysis, algorithm for scheduling a set of project activities	FIFO, LIFO, mathematical study of waiting lines	Economic Order Quantity, ABC Analysis, Mathematical method used for modelling and optimization of storing items in the warehouses	Simulation model, queue of events, experiments (parameters)
Production programme	x				x
Throughput part time		x	x		x
Capacity setup	x		x		x
Resource utilisation	x		x		x
Storage sizes	x		x	x	x
Facility placement and transportatio n system	x	x	x		x

2.2 Layout design methods

As mentioned in Section 2.1.1, physical allocation of resources in the physical space available is a very common task of an industrial engineering department.

„…The concept of FLD is usually considered as a multiple objective problem and its solution is often challenging and time consuming. Facility layout problem usually involves the arrangement of department to optimize the distance travelled by unit of products, people, flow information etc...„

(Fruggiero, Ponte, & Melillo, 2006)

FLD can be managed completely manually or using computer systems. **Manual design** is still a wide used method presently. Methods such as simple machine drawings, paper models, colour pencils or Lego building blocks, have been used. For ergonomic purposes, cardboard boxes are used, replacing real facility objects in the designed workplace. The same approach is often used in urban projects by architects. Manual design is adequate for low complexity systems; however, some disadvantages could be experienced, such as low effectiveness layouts, limitations in results presentation or even few alternatives suggested (Cardarelli & Pelagagge, 1995).

Many methods and algorithms have been developed for FLP, which can be divided into two groups:

- algorithmic methods
- procedural methods

In general, these **algorithmic methods** are based on establishing objectives, processing of input data (quantitative), specifying constraints and designing objective functions. The obtained solutions are then compared, based on the results from the objective functions so that an optimal solution can be selected (Castillo & Peters, 2002)(Peters, 1997)(Cardarelli & Pelagagge, 1995).

Solutions often need **further modifications** involving other layout requirements as realistic department or facilities shapes, material handling system, buffer sizes (WIP storages), physical limitations, fixed facilities or current production systems (re-layout problems), ergonomics, etc. The use of these methods also needs **advanced mathematical knowledge** in modelling techniques. The two mentioned facts lead many companies to decline the full use of these methods (Francis & White, 1974).

Procedural methods use as inputs qualitative and quantitative data sets. The design process is divided into a few phases (steps) that are solved sequentially, several alternatives are designed, and the final layout alternative is chosen. The main advantage of these methods is the possible applicability to different areas and for different tasks. On the other hand, these methods are strictly dependent on user's experience and on the layout design skills (Apple, 1972)(Muther et al., 1973)(Padillo & Meyersdorf, 1997).

Many other methods and principles are used for solving FLP, as **metaheuristic methods** (genetic algorithms, simulated annealing, ant colony, etc.), mathematical modelling or designing in a **virtual reality environment** (Fruggiero et al., 2006) and **interactive design**. Some tools are used as CAD systems (AutoCAD, CATIA, and MS VISIO etc.), database systems or GIS (Geographical information systems – MapInfo, Google Earth etc.). This approach brings 3D working entertainment, walking through layout, or rendering of realistic animation (Whitman, Jorgensen, Hathiyari, & Malzahn, 2004).

A detailed description of FLP and historical overview is presented by Chee (2009), Heragu (2006) or Weng (1999), involving a lot of practical examples.

Historical milestones

The history of FLP solved by mathematical methods started in the 50s through the Quadratic Assignment Problem definition (QAP). During the following decades FLP has been solved by many approaches and in different ways, as shown in Figure 17 (www.ipaslovakia.sk, 2011)(Singh & Sharma, 2005)(Canen & Williamson, 1996).

The following summary shows some important works concerning FLP:

- 1957 - **Mathematical modelling** (Quadratic Assignment Problem, QAP) introduced by Koopmans & Beckman (1957), as a way to get an optimal solution for the FCP
- 1973 - **Systematic Layout Planning**, procedural method (Muther et al., 1973)
- 1979 - Multiple objectives in layout solutions (Rosenblatt, 1979)
- 1985 - Facility layout by analysis of **clusters** (FLAC) (Scriabin & Vergin, 1975)
- 1985 – **Genetic algorithm** (GA) for arrangement of production cells based on part family, (Green & Al-Hakim, 1985)
- 80th - **Artificial Intelligence** (AI), Intelligent Facilities Layout Planning System (IFLAP5) (Kumara, Kashyap, & Moodie, 1988)
- AILAY (AI based layout which is based on intelligence of evaluation and search process in knowledge-based system (Shih, Enkawa, & Itoh, 1992)
- 1992 - **Graph theory formulation** for FLP by Leung (1992)
- 1992 – Design of layouts involving uncertainties (Palekar, Batta, Bosch, & Elhence, 1992)
- 1998 - **Simulation**, incorporation of many requirements into FLP solution (Eneyo & Pannirselvam, 1998)

2.2.1 Algorithmic methods overview

Figure 17 shows the classification of algorithmic planning techniques and methods used for solving **Facility Layout Problem (FLP)**. These methods are involved in the CALD applications (*Computer-Aided Layout Design*).

Analytic solutions for FLP can be formulated according to *Quadratic Assignment Problem* (QAP) as a model in discrete optimization which works by enumerating different layout configurations (alternatives) until the best one is obtained. This strictly mathematical procedure considers that "m" locations are available for "n" departments. QAP is an NP-hard problem (Sahni & Gonzalez, 1976) that implies to be computationally impractical for problems involving more departments due to the increased number of combinations (Partovi, 1992).

Also, these methods couldn't involve all input factors and models must be simplified. Due to these reasons, many **heuristic** methods have been developed to solve FLP. Heuristic methods can provide good sub-optimal solutions in acceptable time. Therefore, these methods have been used in the industrial environment. Two basic approaches are used:

- Exchange methods
- Constructive methods

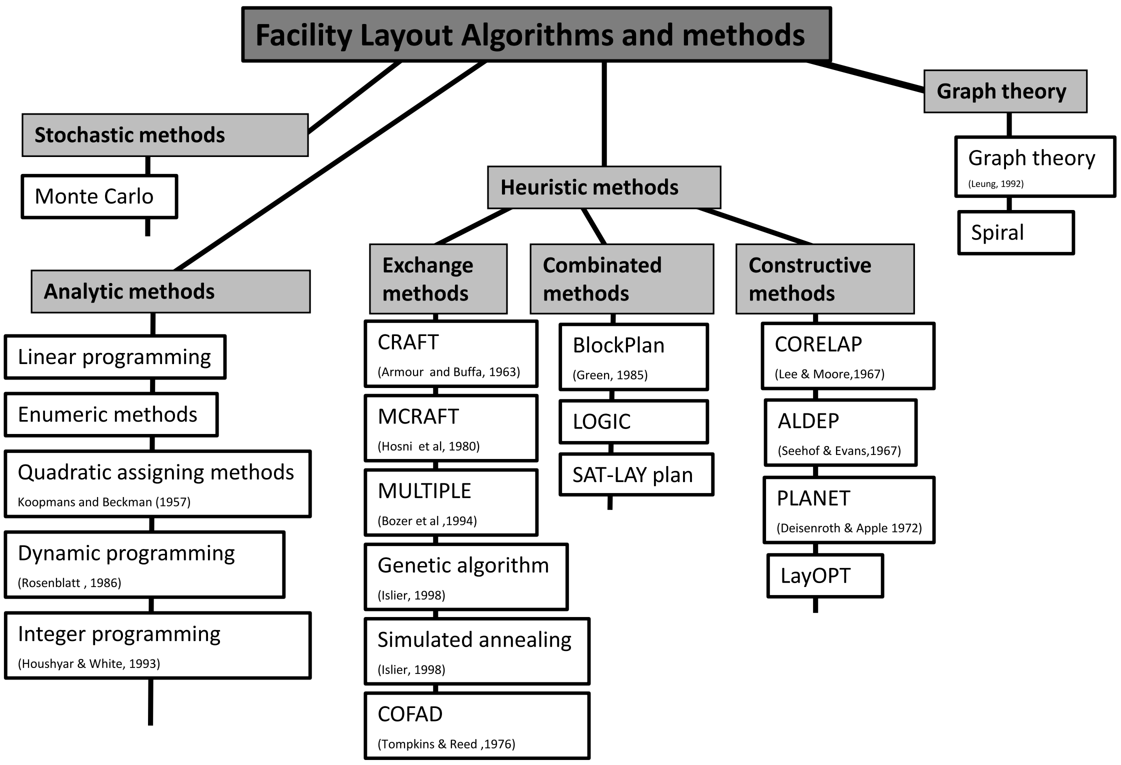


Figure 17 – Summary of algorithmic FLP methods

Exchange or improvement methods use initial block layout usually designed manually by the user himself or developed by constructive methods. It is improved by sequentially exchanging pairs of departments to achieve the minimum of the given objective function (Bozer & Meller, 1994).

The objective function could correspond to minimize transportation costs, minimize empty space between workstations, etc. The most popular are CRAFT, COFAD or MULTIPLE.

Constructive methods generate layouts based on the relationship chart (see more in Section 2.1.1) describing relationships between all workstations or departments (Hassan & Hogg, 1991).

The most popular are CORELAP, BLOCKPLAN or ALDEP. These methods are used in the "*Systematic Layout Planning*" (Section 2.2.2) that is a generic approach of FLP.

These methods are compared to one known real facility problem (Wilsten & Shayan, 2007); Gómez et al. (2000) compare GA and CORELAP approaches.

In **Graph theory**, departments are represented by nodes, and then a planar graph is developed, based on adjacent departments. From this graph, a block layout is generated.

Ferrari (2003) describes a multi-criteria approach (material flows and relationships) with the use of integration solution (MS Access database and MS VISIO for CAD drawings) as "*Logistic and Re-layout Program*" (Ferrari, Pareschi, Persona, & Regattieri, 2003).

A short description of the most significant algorithms and methods mentioned in the previous section is presented below. A lot of information and many examples are available at Concordia University web pages (www.ciise.concordia.ca, 2011).

Some of the algorithms (CRAFT, CORELAP, MULTIPLE, ALDEP etc.) use a matrix for the layout representation (Bozer & Meller, 1994). Each element of the matrix corresponds to a grid square and each department size is expressed by an appropriate number of squares.

a) <u>CRAFT</u>

CRAFT (*Computerized Relative Allocation of Facilities Technique*) is one of the most popular algorithms used for optimising a plant layout. CRAFT conducts a **pair-wise exchange** to develop a layout (Buffa & Armour, 1964), (Hicks & Lowan, 1976).

Input data include the specification of departments (size, shape - as discrete representation), flows between departments (material flows, frequency of movements etc. - as *from-to* chart) and cost (per unit load or per unit distance) and **initial layout** sketch.

The aim is to find the **minimum total cost** (travelling/load) of all moving items (Eq. (1))

C: Total Transfer Cost
i,j: indexes of departments
f_{ij}: Transfer flow From i To j
c_{ij}: Cost of Transfer From i To j
d_{ij}: Distance From i To j (From Centroid)

$$\boldsymbol{Minimize\ C} = \sum_{i=1}^{n} \sum_{j=i+1}^{n} f_{ij} c_{ij} d_{ij} \tag{1}$$

Distances are calculated as the difference between the coordinates of department centroids (Δx and Δy). It can be rectilinear - Eq. (2) or Euclidean Eq. (3).

$$d_{ij} = |\Delta x| + |\Delta y| \tag{2}$$

$$\boldsymbol{d_{ij}} = \sqrt{(\Delta \boldsymbol{x})^2 + (\Delta \boldsymbol{y})^2} \tag{3}$$

To calculate the distance between departments, the centroid of each department is determined using equations (4) and (5). The calculation of a department centroid is simple because each department consists of given square units; therefore the centroid is a sum of square centroids of each unit.

i: index of units
A: area of square unit
Cx,Cy: x, y centroid of unit
n: number of units in department

$$X = \frac{\sum_i^n Cx_i A_i}{nA} = \frac{\sum_i^n Cx_i}{n} \tag{4}$$

$$Y = \frac{\sum_i^n Cy_i A_i}{nA} = \frac{\sum_i^n Cy_i}{n} \tag{5}$$

In Figure 18, there is a description of CRAFT heuristics. It is based on a selection of pairs of departments that can be exchanged – **adjacent** departments (see example in Figure 19, department F and C) or with the **same size** (departments B and E). Departments in each pair are switched and total transportation costs are calculated. From all the possible *Pairs*, it is chosen and implemented into the layout the one with the greatest reduction on transportation costs. These exchanges and implementations are done until no more reduction is possible (Smutkupt & Wimonkasame, 2009).

Department shapes are not restricted to rectangular shapes and in the most advanced methods it is possible to use dummy departments to represent fixed areas in the layout. The final layout score depends on the initial layout because CRAFT method is path-oriented. It is convenient to use several different initial layouts.

The solution can lead to irregular shapes of departments or facility layout; the final layout must be finished manually.

This algorithm considers only costs between departments. There are missing factors as total time in system, waiting time, or resources utilization. The disadvantage of CRAFT algorithm is that it starts with arbitrary initial layout and therefore, there is no mathematical certainty of getting optimal solution (Carrie, 1980).

```
Initial layout
    Calculate transportation cost for the layout (Department centroids (Eq. (4),(5) ), Distance
    between centroids (Eq. (2) or (3)), then calculate (Eq. (1))

    Repeat
        Select Pairs = Same Size Department OR Neighboured Department

            For each pair in Pairs
                Switch pair in layout
                Calculate transportation cost for the layout (Department centroids (Eq.
                (4)(5)), Distance between centroids (Eq. (2) or (3)), then calculate (Eq.
                (1))
            next

        Implement the departmental exchange that offers the greatest reduction in
        transportation cost

    Until no exchange is able to reduce the transportation cost
Final layout
```

Figure 18 – CRAFT heuristic

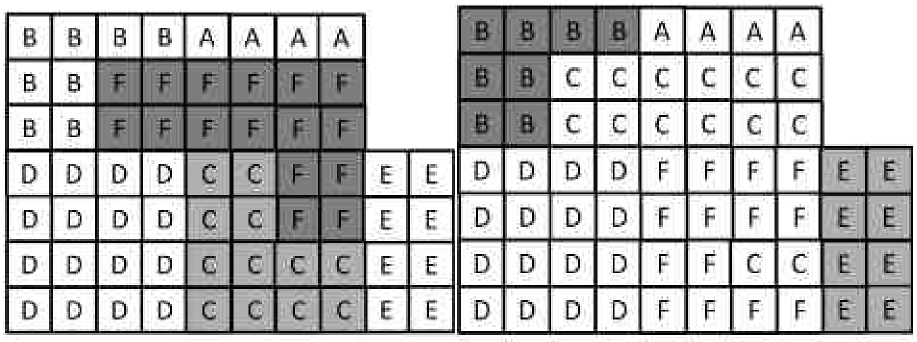


Figure 19 – Department exchanges in CRAFT

Modifications of CRAFT algorithms are **mCRAFT** (Hosni, Whitehouse, & Atkins, 1980) **Optimum Sequence** (Heragu, 2006) or **MASS** ("Modified Assignment") that is an extension of CRAFT by a Hungarian assignment algorithm (Khoshnevisan, 2003).

b) CORELAP

CORELAP (*Computerised Relationship Layout Planning*) is a construction algorithm that generates layouts based on relationships (R. C. Lee & Moore, 1967).

It is an adjacency-based method, as data input is used in the relationship chart with the following closeness rating values A=125, E=25, I=5, O=1, U=0 and X=-125 (see more in Section 2.1.1). Different values can be used based on relationship importance – e.g. A (1024), E (64), I (8), O (1), U (0) and X (-1024). The final layout is sensitive to these weight values. The relationship chart allows merging qualitative and quantitative factors together.

There is an effort to achieve maximum value of the layout evaluation that is based on the adjacent departments (Eq. (6)). Layout alternatives can be compared using a score based on achieved relationships. Equation (7) describes efficiency score for a system without X and Equation (8) with X relationships having negative score values when department is adjacent.

z: layout evaluation
n: number of departments
f$_{ij}$: relationship value between department i to department j
x$_{ij}$: =1 if department **i** and **j** are adjacent and relationships are AEIOU, 0 if not.
=1 if relationship between departments **i** and **j** is X and departments are not adjacent.

$$\textbf{maximize}\, z = \sum_{i=1,}^{m} \sum_{j=1}^{m} f_{ij}\, x_{ij} \tag{6}$$

$$z = \frac{\sum_{i=1}^{m} \sum_{j=1}^{m} f_{ij} x_{ij}}{\sum_{i=1}^{m} \sum_{j=1}^{m} f_{ij}} \tag{7}$$

$$z = \frac{\sum_{(i,j)\in F} f_{ij} x_{ij} - \sum_{(i,j)\in F} -f_{ij}(1 - x_{ij})}{\sum_{(i,j)\in F} f_{ij} - \sum_{(i,j)\in F} -f_{ij}} \tag{8}$$

CORELAP heuristic consists of two phases:

- Department selection (Figure 20)
- Department placement (Figure 21)

Department selection procedure:

In the first phase, a sequence of departments is established for the sequence of placement into the layout. The selection of the departments entering the layout is based on *Total Closeness Rating* (TCR) for each department (Eq. (9)). TCR is the sum of the numerical values assigned the closeness relationships between the department and all other departments.

$$TCR = \sum_{j=1, i\neq j}^{m} w_{ij} \tag{9}$$

The first department is selected and placed into *Sequence* based on the highest TCR value. In case of a tie, the department with more A (then E, I etc.) relationships is selected. If this department has X relationships with the non-placed department, this department is placed in the last position of the *Sequence*. Other departments are selected and placed into *Sequence* based on the strongest relationships with already selected departments. The strongest relationships are based on the number of A (then E, I etc.) relationships. In case of a tie, the department with the highest TCR is selected. The procedure continues until all departments have been placed into *Sequence*.

Ties can be solved by different ways – by the highest TCR (described above), by the largest area of departments, by a random department selection, etc.

```
Calculate TCR (Eq.(9))

Repeat

    Case of first department:
        Select Department = max[TCR(Departments)]
            (in case of tie, department with more A relationships (E, I etc.) is selected)
        Sequence(„first empty position“) = Department

            If Department has X relationship with any other non-placed departments then
                (In case of ties:
                xDepartment=min[TCR(Departments)]
                Sequence(„last empty position“)= xDepartment
            endif
    Case of the rest of departments:

    last_placed = Department

    Select Department= strong relationship with last_placed

    (in case of tie, choose department with highest TCR)

    Sequence(„first empty position“)= Department

        If Department has X relationship with any other non-placed departments then
             (In case of ties:
            xDepartment=min[TCR(Departments)]
            Sequence(„last empty position“)=xDepartment
        endif
Until all departments are in Sequence
```

Figure 20 – CORELAP heuristics - Department selection

Department placement procedure:

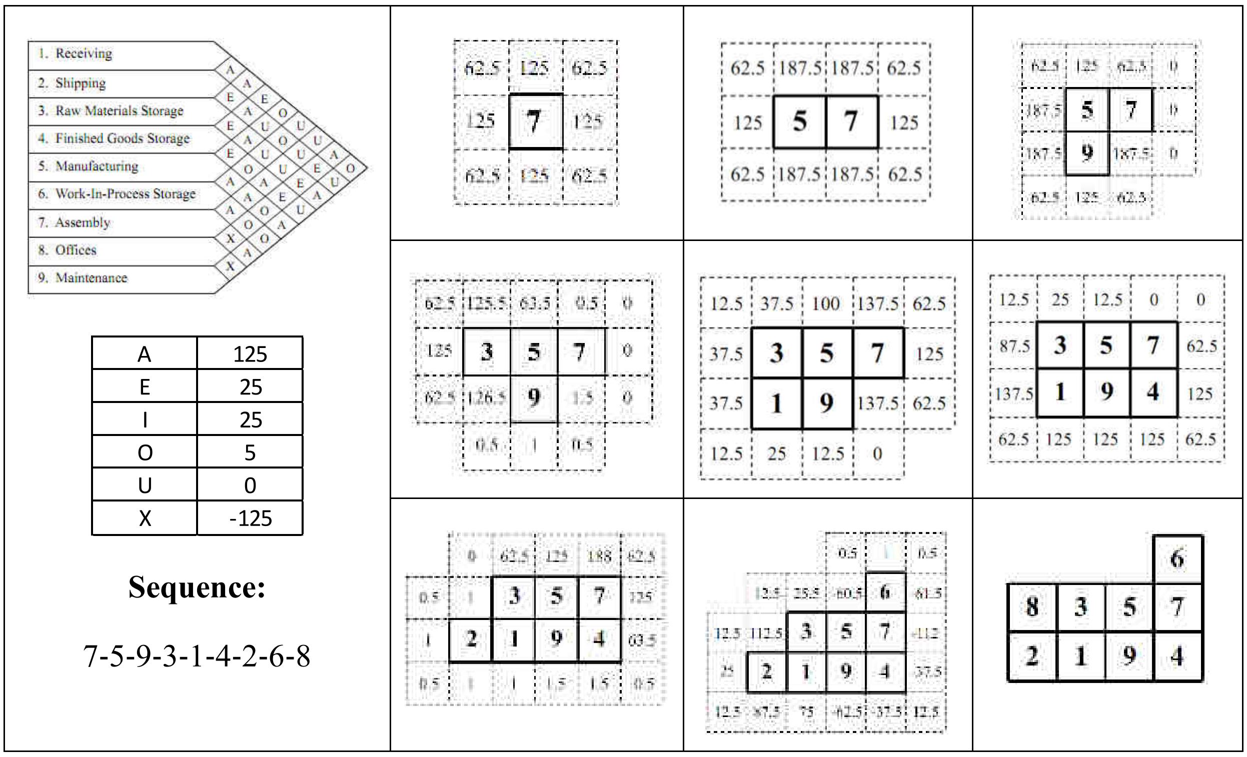
It is based on the established *Sequence*. Departments are represented by an appropriate number of square units that are placed into a grid. Each space of a grid can be evaluated by the placing rating (PR). PR is a sum of the weighted closeness ratings between the departments entering the layout and its neighbours (Eq. (10)). In some CORELAP extensions, the adjacency is divided into full (touch by edges) and corner touching, with different coefficients.

PR: placing rating
w: weighted closeness ratings
k: index for department
n: number of different department in neighbour
i: index of current department

$$PR = \sum_{k}^{n} w_{ik} \tag{10}$$

```
Case of first department:
        Place All units of Department in the central region of layout
Repeat
    Case of the rest of departments:
            Calculate PR score (Eq. (10)) for all possible location around current layout
            Place All units of Department into positions with max (PR)
            (in case of tie, place in clockwise order with at the western edge)
Until all departments units are in layout
```

Figure 21 – CORELAP heuristics - Department placement



A	125
E	25
I	25
O	5
U	0
X	-125

Sequence:

7-5-9-3-1-4-2-6-8

Figure 22 – CORELAP – placing of departments (adapted from (www.ciise.concordia.ca, 2011))

Final layouts alternatives often result in irregular building shapes (not just rectangular) because CORELAP does not take the building shape into account. Therefore, the rearrangement of some department units by the user is needed.

c) <u>ALDEP</u>

ALDEP (*Automated Layout Design Program*) (Seehof & Evans, 1967) is similar to CORELAP, with the same objectives and requirements. The differences are in the random selection of departments. The first department is selected randomly. The next department is selected from those departments that have more A (or E, I, etc.) relationships with the one already selected.

This procedure is repeated until all departments are selected. Then, department units are placed in the layout into vertical sweep pattern, with the width that the user defined, as it is shown in Figure 23. Department units are placed into vertical sweep pattern in the width defined by the user. The example shows placement of sequence (and department size) D8-B6-A3-C4 for width=1 and 2.

This complete procedure is repeated at least 20 times. The result alternatives are evaluated and the best is chosen.

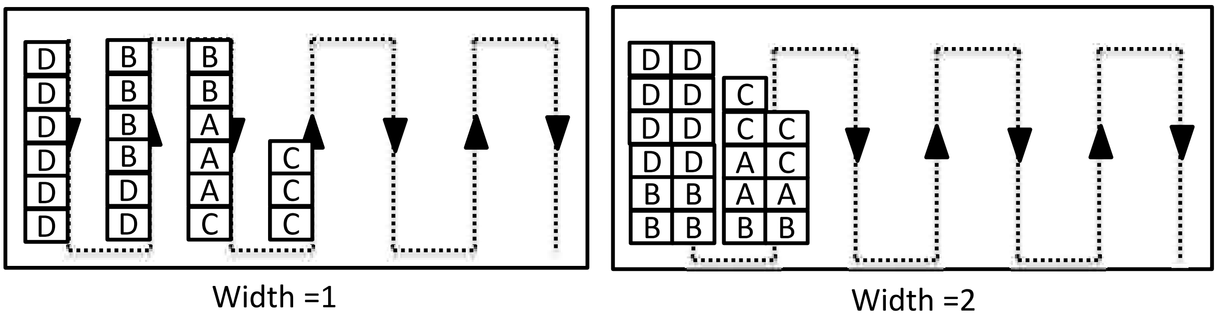


Figure 23 – ALDEP

d) <u>MULTIPLE</u>

MULTIPLE (*Multi-floor Plant Layout Evaluation*) is a construction and improvement algorithm, similar to CRAFT. The differences are that in MULTIPLE there are possible exchanges of non-adjacent departments with unequal sizes and departments are not restricted to rectangular shapes (Bozer, Meller, & Erlebacher, 1991).

Input data for this algorithm is a list of departments and their sizes, sequence (vector) of departments and initial layout frame shape.

Algorithm uses *Space Filling Curves* (SFC)(Parker, 2008) to reconstruct a new layout after each iteration in which a pair of departments is exchanged. These curves connect all grids in the layout, in vertical or horizontal direction (similar approach as in ALDEP using sweep pattern). Each grid is visited once; the next grid visited is always adjacent to the current grid.

Figure 24 shows MULTIPLE algorithms, and the layout frame shape is filled by SLV curve. Then, departments are placed, based on the layout vector that can be initially given randomly. The SFC is followed while the required number of grids for each department is not reached. In this example, inputting sequence (size) is D(16)-B(8)-A(4)-F(16)-E(8)-C(12). Departments are exchanged one by one to achieve minimum transportation costs; the principle is the same as in CRAFT algorithm (see Figure 18).

MULTIPLE is used in cases where building shape is irregular - there are fixed areas or building limitations such as walls inside the layout. The fixed departments and obstacles are not visited by SFC, and MULTIPLE captures exactly the initial layout.

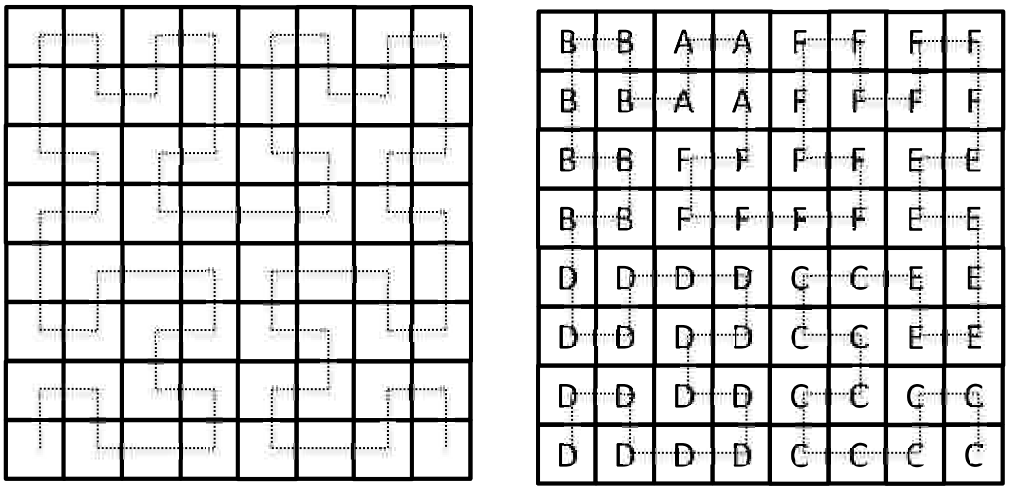


Figure 24 – MULTIPLE

e) <u>Circle method</u>

This method is based on the need of planning material flows that must be as short as possible. Input data is *From-To* chart. The sum of products of material flows and distance is minimum (Eq.(11)).

a_i : distances
G_i: material flows

$$min = \sum_{i=1}^{n} G_i a_i \tag{11}$$

Just the distance between departments is known, not the direction. This distance is given by material flows, based on Eq. (12). Department lies somewhere on the circle with radius a_i with a centre in the first department.

M: scale of layout
G_i: material flows

$$a_i = \frac{1}{G_i} M \tag{12}$$

Using this method, actual department´s locations are not found - only the relative position of departments. This method is not efficient for a higher number of departments or with complex material flows pattern.

f) <u>Coordination method</u>

This method can be used for the placement of central department as storages, etc. where the best location in the current layout should be found. The solution is based on finding the weighted centre of given department (facility) (see Eq. (13) and (14)).

q_i: material flows
x_i, y_i: coordinate of departments
i: index of department

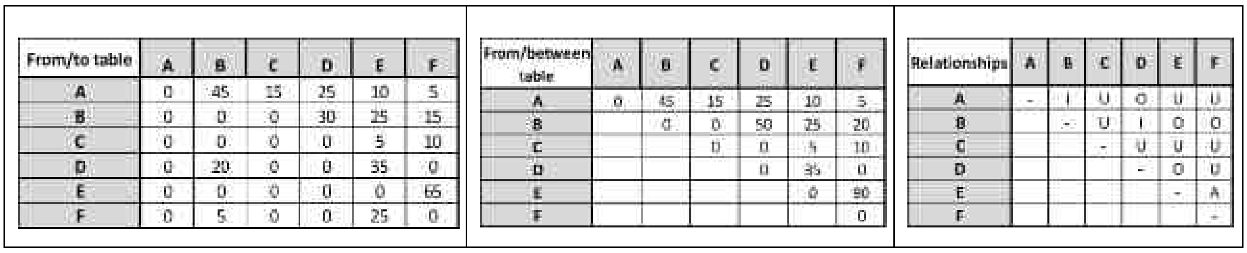
$$x = \frac{\sum x_i q_i}{\sum q_i} \tag{13}$$

$$y = \frac{\sum y_i q_i}{\sum q_i} \tag{14}$$

g) <u>BlockPlan</u>

BLOCKPLAN is a construction and an improvement algorithm (Green & Al-Hakim, 1985). It combines both quantitative and qualitative data to generate several block layouts and their measure of fitness.

Input data are *From-To* Chart and *Relationship chart*. These two inputs can be converted and merged together. *From-To* chart is converted to relationship chart through *Flow-between* as shown in Figure 25. Values of material flows are replaced by relationship values (AEIOU) into five intervals corresponding to five relationships, e.g. "A" relationship is for material flows from 90 till 73, E for 72 till 55 etc.



From/to table	A	B	C	D	E	F
A	0	45	15	25	10	5
B	0	0	0	30	25	15
C	0	0	0	0	5	10
D	0	20	0	0	35	0
E	0	0	0	0	0	65
F	0	5	0	0	25	0

From/between table	A	B	C	D	E	F
A	0	45	15	25	10	5
B		0	0	50	25	20
C			0	0	5	10
D				0	35	0
E					0	90
F						0

Relationships	A	B	C	D	E	F
A	-	I	U	O	U	U
B		-	U	I	O	O
C			-	U	U	U
D				-	O	U
E					-	A
F						-

Figure 25 – BlockPlan (transforming material flows into relationship)

Relationship chart is converted to numerical relationship chart based on closeness ratings (usually by A=10, E=5, I=2, O=1, U=0 and X=-10). For layout evaluation, it is used:

- Distance-based score (Eq. (1))
- Adjacency-based score Eq. ((7) and (8))
- REL-DIST score (flow-between equivalent) (Eq.(15))

REL-DIST score uses numerical **closeness ratings** $\mathbf{r}_{ij}$ based on the relationship chart instead of material flows values. It can be helpful for cases where there is no from-to charts (see Figure 26).

i,j: departments
$\mathbf{r}_{ij}$: closeness ratings
$\mathbf{c}_{ij}$: costs
$\mathbf{d}_{ij}$: distances
z: layout score

$$z = \sum_{i=1}^{m} \sum_{j=i+1}^{m} r_{ij} c_{ij} d_{ij} \tag{15}$$

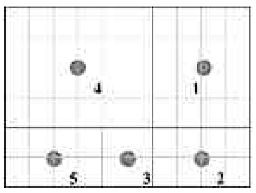
Relationship chart

	D1	D2	D3	D4	D5
D1		A	U	E	U
D2			U	I	I
D3				U	I
D4					A
D5					

Numerical relationship chart

	D1	D2	D3	D4	D5
D1		10	0	5	0
D2			0	2	2
D3				0	2
D4					10
D5					

Layout



Distance matrix based on layout

Distance	1	2	3	4	5
1	-	3	6	5	9
2		-	3	8	6
3			-	5	3
4				-	4
5					-

Total cost matrix

	D1	D2	D3	D4	D5
D1		30	0	25	0
D2			0	16	12
D3				0	6
D4					40
D5					

Figure 26 – BlockPlan (transforming relationships into cost matrix)

h) <u>Graph Theory Block Layouts</u>

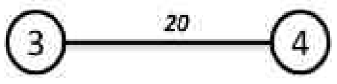
Graph theory (GT) applies an edge–weight maximal planar graph in which the vertices (**V**) represent the facilities and the edges (**E**) represent adjacencies (Foulds & Robinson, 1976) (Kim & Kim, 1985)(Leung, 1992). Graph theory can be used for production systems layout developing (Hassan & Hogg, 1991).

Layout construction is based on a graph theory where relationships are given by weights. Relationships can have a positive value only when departments are adjacent. Other relationships are ignored. In the process of layout construction, dimensional specifications of departments are not considered in the first phases, as well as the length of common boundaries between adjacent departments. Figure 27 contains heuristics definition. Process of schematic layout is shown in Figure 28 as a simple example. It begins by selecting a pair with the largest flows from material flow chart. This pair (department 3 and 4) is placed in the layout space. Process continues by selecting and placing department with highest flows with already placed departments. It is department 2 and in next step department 1. Placing the department 5 is based on the calculation of optimum position inside the possible faces and flows between these departments (face 1-2-3 gives total flows 7, face 1-2-4 gives 9 =optimum or face 1-3-4 with 2).

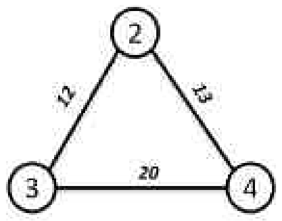
```
Select the department pair with the largest weight, ties are broken arbitrarily.
Repeat

        Select the next department based on the largest sum of the weights with the already placed
        departments
        Place department into bounded region of a graph
        Determine an adjacency graph

Until all departments are placed
Reconstruct a corresponding block layout that consider department sizes
```

Figure 27 – Graph Theory Block Layouts heuristic

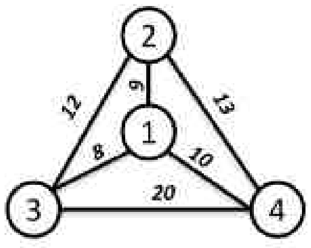
Department flows	1	2	3	4	5
1	0	9	8	10	0
2	9	0	12	13	7
3	8	12	0	20	0
4	10	13	20	0	2
5	0	7	0	2	0

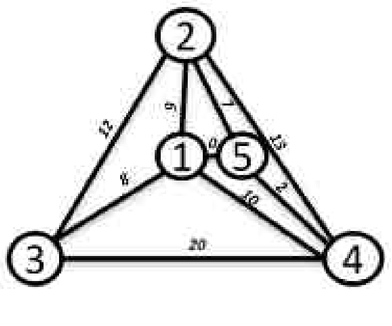


Department flows	3	4	Σ
1	8	10	18
2	12	13	25
5	0	2	2



Department flows	2	3	4	Σ
1	9	8	10	27
5	7	0	9	9





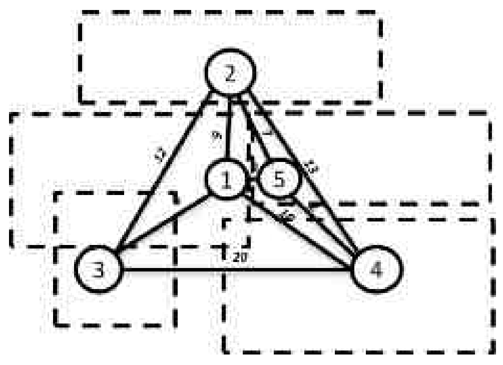
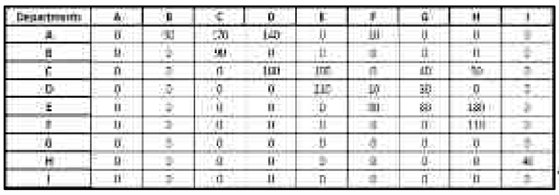


Figure 28 – Graph Theory Block Layout

Using this method, related location of departments is found. Then the block layout considering department sizes (dashed rectangles) is constructed from this schematic layout (see Figure 28, last picture).

i) <u>Triangular method</u>

This method is a modification of the *Graph theory* (see Figure 27). Departments are placed into triangular grids. The aim is to minimize the total material flows between departments. This method is based on the presumption of **the shortest** straight line **connection** and **equal distance** between two **objects**. The condition of equal distances is managed by equilateral triangle. Figure 29 shows an example: *From-To* chart with material flows as input data and constructed layout based on triangular method. In this figure, the numbers in brackets correspond to the sequence of departments placed in the layout (Věchet, 1982).



Departments	A	B	C	D	E	F	G	H	I
A	0	90	170	140	0	10	0	0	0
B	0	0	90	0	0	0	0	0	0
C	0	0	0	100	100	0	40	50	0
D	0	0	0	0	110	10	30	0	0
E	0	0	0	0	0	100	80	130	0
F	0	0	0	0	0	0	0	110	0
G	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0	40
I	0	0	0	0	0	0	0	0	0

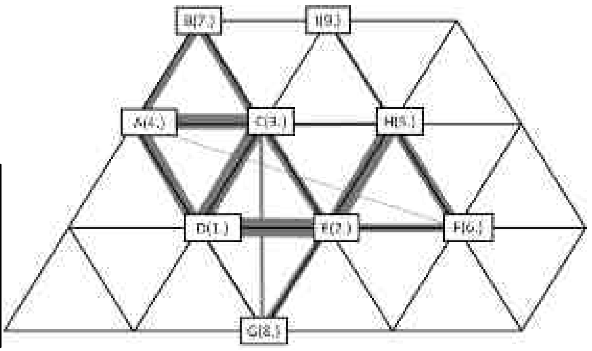


Figure 29 – Triangular method

j) <u>Quadratic Assignment Problem</u>

Quadratic Assignment Problem (**QAP**) is one of the oldest methods for FLP (Koopmans & Beckman, 1957). The aim of QAP is to minimize the materials handling cost by optimum assignment of the departments into location and position. It is a combinatorial problem with a nonlinear objective function (Garey & Johnson, 1979)(Bozer & Meller, 1994)(Urban, 1987).

There are ***n*** departments assigned to ***n*** locations. The objective function finds the minimum for total transportation costs between all departments (Eq. (16)). The required input data are distance table **d** (*From-To* table for all locations) and flows **f** between all departments. Transportation costs are proportional to flows (f) and to distance (d). Departments have the same size and therefore they can be interchanged with each other.

The constraint described in Eq. (17) means that each department ***h*** can be placed exactly on one location ***j***. Equation (18) defines that each location ***j*** gets exactly one department ***h***.

f_{hi}: material flow from department h to department i
d_{jk}: distance between location j and location k
x_{hj}:binary decision variable (=1 if department h is assigned to location j, =0 otherwise)

$$minimalise\ Costs = \sum_{h=1}^{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} f_{hi}d_{jk}x_{hj}x_{ik} \tag{16}$$

With following constraints: (17)
$\sum_{j=1}^{n} x_{hj} = 1$, for h=1 to n

$\sum_{h=1}^{n} x_{hj} = 1$, for j=1 to n (18)

In QAP, departments have the same size. Distances are the shortest distances between centroids.

QAP is an NP-hard problem (Sahni & Gonzalez, 1976). The QAP problem is among the most difficult combinatorial problems. For a long time, problems with 10 departments at most could be solved successfully in real time. The development of computer techniques has been improving this performance – in 1992, QAP problems with 15 departments (Partovi, 1992) could be solved, and in 2002, 30 departments were managed adequately (Anstreicher, Brixius, Goux, & Linderoth, 2002).

Therefore, finding a solution for larger problems is almost impossible by QAP method. Because of this limitation, a great number of heuristic and meta-heuristic methods have been developed.

k) Genetic algorithm

Genetic Algorithms (**GA**) came from the 70's through the work of Holland (Holland, 1992). Many GA applications resolving optimization problems may not achieve optimum solutions but results are very close to the optimum ones. GA involves three basic steps – selection, reproduction and replacement. Each repetition (iteration) is called a generation. GA works with a population of possible solutions, in comparison with other heuristic search methods working usually with a single solution.

There are numerous ways of formulating facilities Layout problems through genetic GA (Askin & Zhou, 1998)(Banerjee, Montreuil, Moodie, & Kashyap, 1992).

GA can be used for some more complex tasks as solving the unequal area layout problem with geometric constraints, improving space utilization (Tam and Chan, 1995)(Wu & Appleton, 2002)(Ahmad, Basir, & Hassanein, 2004). A lot of GAs use a slicing tree structure (Otten, 1982)

(Garces-Perez, Schoenefeld, & Wainwright, 1996)(Wilsten & Shayan, 2007).

2.2.2 Procedural methods overview

Solving FLP by procedural methods is divided into a few phases that are solved sequentially. Several alternatives are designed, and then the final layout alternative is chosen. These methods use as inputs qualitative (*relationship charts*) and quantitative (*From-To* table) data sets.

Procedural methods are often under company confidential knowledge; companies hold it in secret because of their concurrence. First work concerning layout planning methods is from 1950, where basic steps are presented in the layout design, aiming to minimize the distance travelled between work centres and optimization of material handling and flows (Immer, 1950). Reed (1967) suggested a “layout planning chart” that helps to control important phases of layout design and make it clear and simple. Apple (1972) also described a sequence of steps used for factory layout.

The most used procedural method is ***Systematic Layout Planning***, **SLP** (Muther et al., 1973) which is described below in detail. It uses a graphical representation of facilities and departments and builds up a proximity matrix which represents the closeness of each facility based on the relationships.

The great advantage of these methods is, again, the applicability to different areas and tasks. On the other hand, these methods are strictly dependent on the user’s experience and layout design skills.

Systematic Layout Planning

SLP was firstly presented by **Richard Muther** in 1973 (Muther et al., 1973) and third edition named *Systematic Layout Planning Pattern*) (Muther, 1994) has been developed since then.

Many modifications of this original method have also been written (Wiyaratn & Watanapa, 2010)(Chee, 2009)(Weng, 1999). There is also a *Simplified SLP method* (Muther & Wheeler, 1994) for smaller project’s size or for non-complex cases. It has been widely used since its first introduction in many projects and companies.

It formalizes the whole facility planning process as a set of procedures through which each layout project passes. The design of process starts stepwise, from the:

- generation of alternatives
- evaluation
- selection
- implementation

This method is based on the relationship and the space diagrams that allow connecting quantitative and qualitative influences together.

SLP is used because it is a universally-applicable approach almost for any layout planning project. Also, this method supports systematic approach: from general issues to details.

General description

SLP is based on the *Facility Planning* phases (see Section 1.1.2). SLP is usually concerned only into the three last phases, without plant location - that is a quite rare type of problem. Block and detailed layout phases are also often overlapping because some information from detailed layout must be known in the block layout design phase – e.g. required space containing all machinery of given department. It assures that sufficient space and adequate overall dimensions have been provided for each department.

For each of these phases (mainly for designing block layout and detailed layout), a **list of instructions** can be used, as it´s shown in the schema (Figure 30).

Project definition and input data gathering

Firstly, the project aims need to be established. Based on them, the gathering of required input data can start.

Input data are often called as **PQRST** (**P**roduct – **Q**uantity - **R**outing - **S**ervice - **T**ime). Table 7 contains a description for each item.

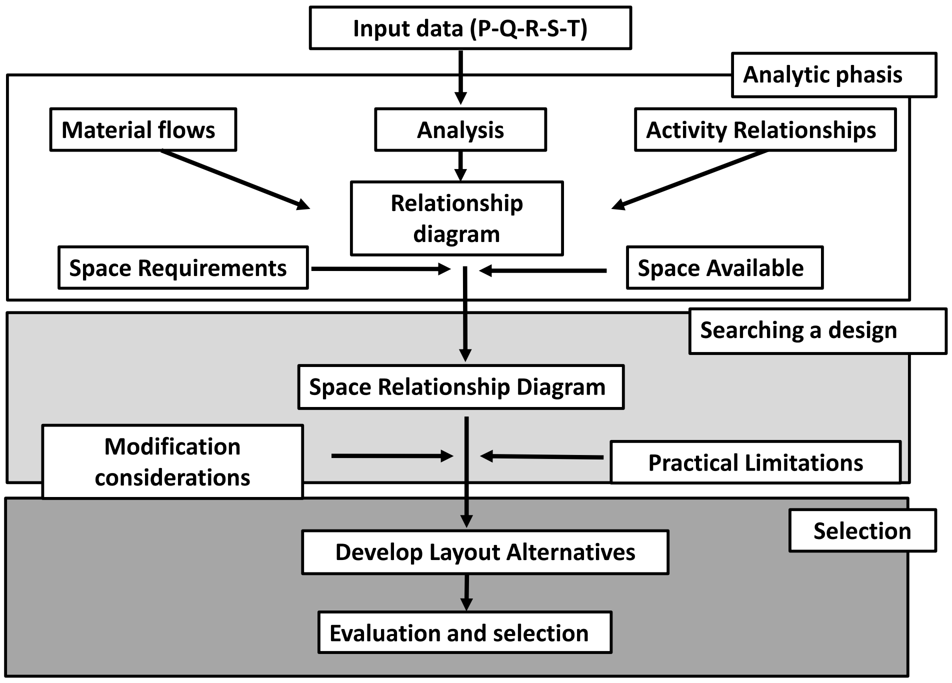


Figure 30 – Systematic Layout Planning schema

Table 7 – PQRST

Sign	Data class	Description
P	**Product**	material or service information
Q	**Quantity**	volume for each product
R	**Routing**	movement information of each item, table of production processes, operation sequences, process sequence
S	**Service**	setup, material handling, supporting services, etc.
T	**Time**	time information for each process or operation

Analytic phase

Based on the PQRST data, analysis can be processed to get to know more information about the system. There is a set of analysis and data input processing; not all of them are important for each type of system (project). The aim of this analysis is to prepare data and information for relationship chart.

Analysis frequently used in the SLP method:

- **P-Q analysis** – see Section 2.1.1 - P-Q analysis (Pareto analysis)
- **Cluster analysis** – see Section 2.1.2 - Cluster analysis
- **Material flow** analysis – see Section 2.1.2 - Flow analysis and *From-To* chart

Relationship chart and diagram

Based on previous studies and on a list of important factors for a system, it is possible to build a relationship chart and diagram, as it was described in Section 2.1.1.

Space Relationship Diagram

This phase of the SLP is based on the established relationship diagram and on departmental space specification consisting of the total required space of given department as:

- space for production equipment (machines, tools, maintenance etc.)
- space for storages
- space for material handling (aisle, manipulation)
- personnel activities

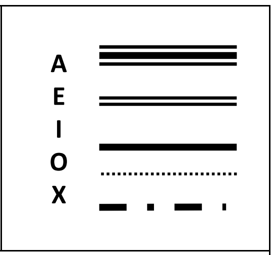
The important issues in department size specification are costs for renting space or building costs. Based on the required space sizes and relationship diagram, it is possible to design **space relationship diagram** for complete plant layout. Departments are replaced by simple blocks (usually square units).

The purpose of the space relationship diagram is to combine established **space areas with**

the activity relationship diagram. The aim is to achieve schematic layout composed from simple rectangular blocks representing departments and representing department closeness or undesirable proximity. Figure 31 shows the construction of space relationship diagram for given relationship chart (see Figure 16) and department sizes. In the first step, departments are placed into a space based on a relationship diagram and then, they are arranged based on a factory frame, shape limitation and other constrains, but still considering relationships. In this example, it appears one free unit between department A and D that can be linked to one of them.

Department	A	B	C	D	E	F
A		X	U	E	U	O
B			A	U	X	I
C				U	U	U
D					U	A
E						A
F						

Department	Size
A	64
B	30
C	40
D	10
E	16
F	35



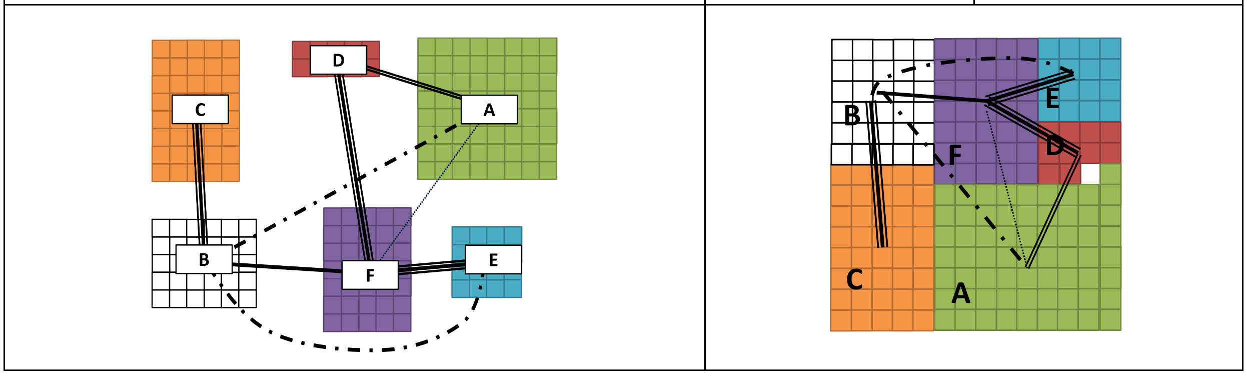


Figure 31 - Space relationship diagram

The construction of the space relationship diagram for factory layout is similar in the facility layout design and both phases influence each other by **feedbacks**.

Developing alternatives of Block layouts and Detailed layout

The development of **block layout** alternatives is based on the space relationship diagram. This diagram is modified by constraints as building limitation, space limitation, current (fixed) production system, etc. This schematic layout consists of blocks without any inner details. Figure 32 (left image) shows this type of layout and, on the right side, there is a detailed layout of the same system (Tompkins et al., 2010).

Blocks of space are developed and positioned according to their relationships defined in the relationship chart.

In this phase, several different layout alternatives are designed and, after that, the optimum candidate is selected as a final block plan layout.

On the basis of block layout and facilities requirements, alternatives of detailed layout are developed. There are arrangement of machinery, equipment and other objects. In Figure 32, the right image represents the detailed layout. In this phase, for the established type of layout, methods such as cluster analyses, P-Q, could be helpful.

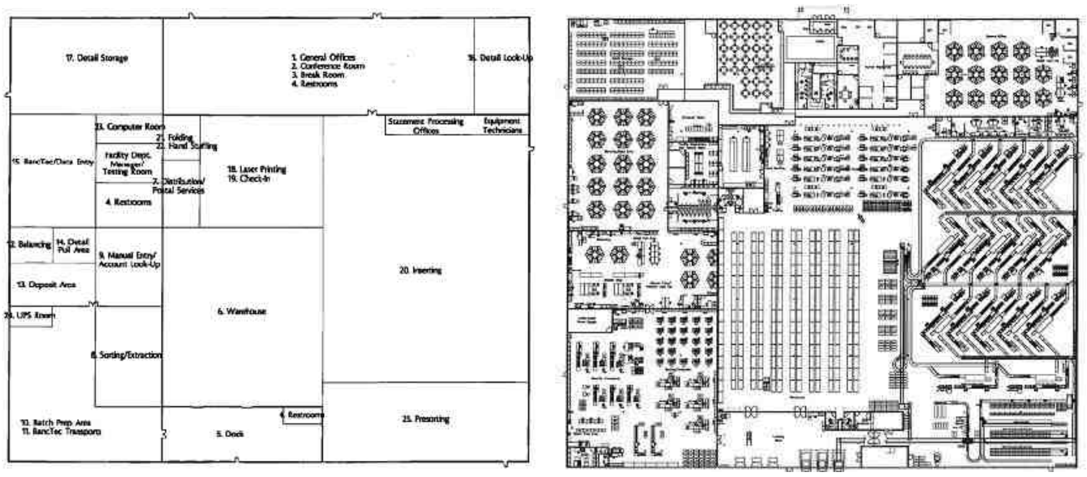


Figure 32 - Block layout and Detailed layout (adapted from Tompkins et al., 2010)

Selection and implementation phase

The best alternative is chosen from the designed alternatives (candidates). It is usually done by a team and its own experience or by the best score (minimum material flows or adjacency of departments given by relationship chart).

The implementation consists of preparing detailed technical drawings; performing building changes in installation of facilities, training workers and starting production.

2.2.3 Conclusion of FLP methods

A large amount of work been published on FLP and layout optimization techniques. These methods usually use as input data set *From-To chart* or *Relationship chart*, with very few papers considering material flows and relative closeness simultaneously.

Algorithmic methods usually cover single criteria problems, **inadequate** for real situations involving more complex relationships among facilities that need multi-criteria approach). Most of them use **constraints** that are too explicit and rigorous for real problems or daily layout tasks or, on the other hand, constraints are too simplified (e.g. space sizes and complex shapes can be replaced by rectangle blocks). Therefore, generated alternatives are often too far for a direct practical use and it needs user action (Tam & Li, 1991) (Sly, 1993).

Procedural methods consider a large number of factors. The process of layout design continues with well-defined steps but these methods need a higher knowledge of industrial engineering, technology, etc. to avoid designing low performance systems.

„...There is no commercial package that will suit all the needs, partly due to the difficulty of the problem, but more due to the fact that facility layout is a combination of Science and Art...„

In fact the statement above corresponds to a common conclusion in many of the publications. Facility Layout is a combination of Science and Art (Tompkins et al., 2010)(Meyers & Stephens, 2005).

2.3 Production systems design software tools

In this section, software tools that help industrial and layout engineers with systems design are described. There are a great number of software tools and it is possible to divide them into several groups based on functions provided. The most important tools are: ***Database*** (DB), ***Discrete Event Simulation*** (DES) and ***Computer Aided Design*** system (CAD). For each type of tools, a typical set of tasks and functions are mentioned.

These tools and applications are usually used in projects with different levels of integration: completely separate and independent, with low level of integration up to completely integrated systems. In this chapter, there is a summary of the State-of-the-art of integration and mentioned systems with high level of integration – ***Digital Factories*** (DF).

2.3.1 Database system

a) Database overview

Database system (DB) is a helpful tool for organizing, storing, and retrieving data required for a project. Based on that data, it is possible to make an analysis to sort information about the system and its identification and also to discover some hidden information - **data-mining**.

Database management systems (DMS) are software tools for managing DB.

Structure of DB

Each DB consists of a collection of data that is organized in one or more *tables*. Tables are basic elements of the DB. Each table has a unique name, a defined data-structure and it contains the user's data.

The elementary term of the DB theory is **relation**. It can be a table which consists of columns and rows. **Columns** correspond to properties (*attributes*) with a specific data type as *string*, *integer*, *date* etc. **Values** are recorded in the appropriate fields in the cross columns-rows. A set of relations (tables) build a whole DB according to the relation structure (Powell, 2006)(Hellerstein, 2007).

The structure of a DB is a key issue. A well-designed DB is more transparent, faster in work and does not contain duplicate values. For this purpose, it is very helpful or even required to establish one property (column) in the relation as a ***primary key***. The primary key has a property that is a value uniquely determined. It allows a unique data combination from different tables of the database and it is the main approach of relational DB. ***Relationships*** between tables are created by using a field from one table and relating it to another table. When the primary key is used to establish relationships in a different table (*child table*), it is called a ***foreign key***.

There are three types of **relationships**: *one-to-one*, *one-to-many* and *many-to-many* relationships. One-to-one relationship means that for each record in one table, there is only one matching record in the related table. One-to-many relationship states that for each record in one table, there can be any number of matching records in the related table. Many-to-many relationship means that records in both tables can have any number of matching records in the other table. Usually, this relationship cannot be directly created in some database tools (e.g. MS Access). However, it is possible through *mapping tables* where one-to-many relationships of two tables are joined (Alexander, 2007).

Indexing is a special technique for improving the database performance. The simplest form of index is an ascending/descending sorted list of values that can be searched using a *binary search algorithm* (Donald, 1999) with a reference to the entry location. Indexes can speed up searching in DB but they consume space and they must be updated every time when data are changed, which requires some maintenance. Indexing affects performance of DB but not results.

SQL language

Structured Query Language (SQL) is a database language, developed in the early 1970s by IBM. It was accepted as standard by ANSI (American National Standards Institute) in 1986 and by ISO standard (and International Organization for Standardization) in 1987. It is supported by most of the database systems and it is the most used database language (Beaulieu, 2009).

SQL provides:

- access to data saved in the databases (insert, query, update, delete data)
- database structure changes (create, modification of tables and database access)

Example of SQL prescription and code:

```
SELECT [ALL | DISTINCT] columnname1 [,columnname2]
FROM tablename1 [,tablename2]
[WHERE condition] [ and|or condition...]
[GROUP BY column-list]
[HAVING "conditions]
[ORDER BY "column-list" [ASC | DESC] ]
```

```
SELECT * FROM Book WHERE price > 100.00 ORDER BY title;
```

The SQL language is sub-divided into several language elements as queries, clauses (*WHERE* …), expressions (*total_count+1*) or predicates (*name = 'USA'*).

SQL language is well-known and additional information can be found in many sources (Weinberg, Groff, & Oppel, 2009)(www.w3schools.com, 2011)(www.sql.org, 2011).

Alternatives to SQL

Although SQL is the most widespread database language, there are also alternatives as Query Language (4D QL), Datalog, HTSQL (URL based query method), ISBL, Java Persistence Query Language (JPQL), LINQ, Object Query Language or XQuery. (see full list at (http://en.wikipedia.org/wiki/SQL, 2011).

b) Database management systems

Database Management Systems (DMS) are software applications used for storing, searching, updating data, database administration, and security of users' access to DB. These operations are usually done through Input/Output forms, which facilitate the user´s work. There is a great number of DMS in the market. The most popular are MySQL, Oracle, PostgreSQL, Microsoft Access, SQL Server, FileMaker, Firebird, Sybase, dBASE, Clipper, FoxPro etc. It is not relevant to mention the complete properties of them in this work.

Open Database Connectivity (ODBC) driver helps data integration between two different DBs.

c) Database in production systems (ERP)

Typical usage of database systems in production systems design is in Enterprise Resource Planning systems (ERP). ERP usually involves a complete data set of company organisation, manufacturing, accounting, finance, external customers' data, etc. Activities and data operation with these data can be done automatically by ERP system. ERP in the production system analysis, facility design or re-layout can be helpful as data source; it can involve information as scheduling, capacities, bill of materials, manufacturing process and flows, product lifecycle management, etc. Based on these data, simulation models of production models are easy to build.

The most widespread ERP packages are SAP, Oracle, Sage Group, Microsoft Dynamics and SSA Global Technologies.

2.3.2 Discrete Event Simulation tools

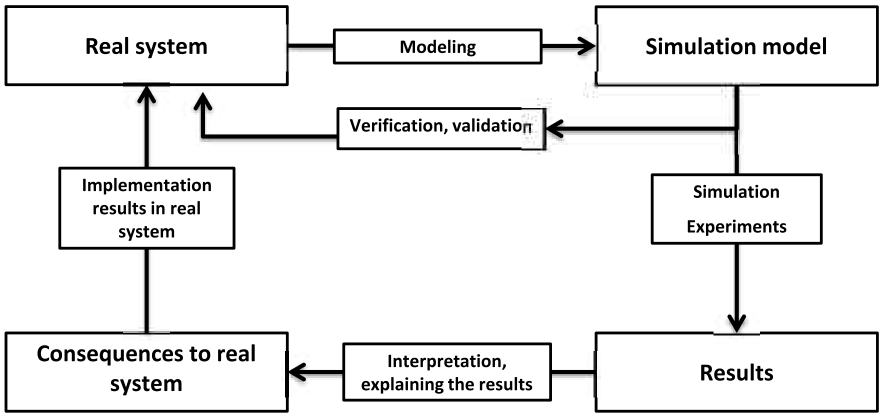
Discrete Event Simulation (DES) is a dynamic modelling methodology. DES manages a queue of object´s **events** (sequence of events) sorted by the simulated instant they should occur. Simulation programme (*simulator*) reads this queue and, in the appropriate time, it triggers the given event from the queue as each event is processed. It is possible to study the interactions between system objects and dynamics of a system over time. One of the most convenient properties is that the simulation running couldn´t run in real time rate but the simulation time can jump a time period between consecutive events(Banks, Carson, Nelson, & Nicol, 2005).

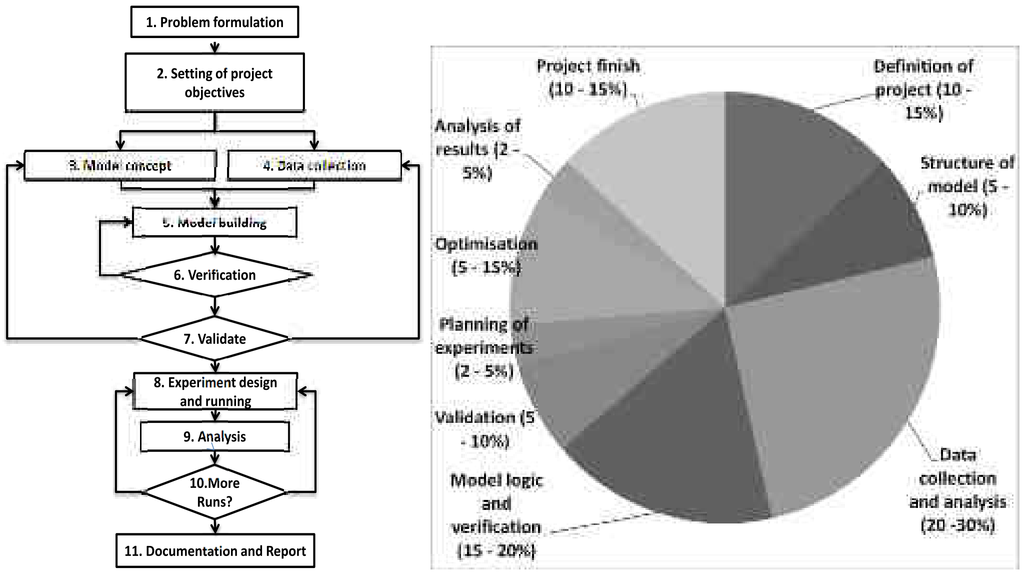
DES can be used for describing, running *experiments* and for the optimization of real processes by a virtual *simulation model*. The simulation allows making experiments with a virtual model of a real system. Changing the real system for experiments can be expensive, dangerous or even impossible if it deals with a planned system. In the case of a planned system, it is possible to identify probable problems as design shortfalls and bottlenecks, before building or modifying a real system.

The received information can be used as feedback in the real system, as shown in Figure 33.

Using simulation, it is possible to get information about the resources **states** (working, waiting, transportation, breakdown, service etc.).

It is convenient to use simulation in cases where it is not possible to make a simple analytic model, spread sheet model or envelope calculation due to the requirement of more accurate analysis of the real system. DES is applicable to real systems consisting of objects that can be defined and characterized as well as their interactions or interdependences. The real system is regularized, not chaotic and out of control (Manlig, 1999), (Fu, Glover, & April, 2005).


Real system
Modeling
Simulation model
Verification, validation
Implementation results in real system
Simulation Experiments
Consequences to real system
Interpretation, explaining the results
Results
Figure 33 - Simulation principle


1. Problem formulation
2. Setting of project objectives
3. Model concept
4. Data collection
5. Model building
6. Verification
7. Validate
8. Experiment design and running
9. Analysis
10.More Runs?
11. Documentation and Report
Project finish (10 - 15%)
Definition of project (10 - 15%)
Analysis of results (2 - 5%)
Structure of model (5 - 10%)
Optimisation (5 - 15%)
Planning of experiments (2 - 5%)
Validation (5 - 10%)
Model logic and verification (15 - 20%)
Data collection and analysis (20 -30%)
Figure 34 - Model for building phases and the required time for each phase

a) <u>Simulation model</u>

A simulation model is a descriptive model of processes in the system. This model can be configured by a set of **parameters** representing different system configurations. It is used for making experiments, evaluating and comparing system alternatives. Evaluation, comparison and system analysis are the main reasons for doing simulation. Prediction of system performance and identification of system problems and their causes are the key results. In Figure 34, there is a flowchart of simulation model building and also the estimated time requirements for each phase (Manlig, 1999). Simulation projects start with the establishment of aims, scope and model boundary. Then, the conceptual model is sketched and the required data is collected and processed. Based on that simulation, the model is built. Two important phases must be passed – *verification* and *validation*. Verification is a check of the model functionalities, whereas validation is comparing the simulation model with the real system behaviour and properties. After that, this model is prepared for running experiments and result analysis. Reporting and making documentation are very important phases to perfectly understand presumed consequences (effects) into the real system. It can be supported by animation of processes.

b) <u>Model simplification</u>

A very important issue in building the model is the simplification. In many cases, it is not necessary to simulate all objects (resources) appearing in the real system but only those that are significant in the project scope. Also, it is not necessary to describe all objects properties but only those that are mandatory for the established project aims.

In fact, very detailed models could lead to some loss of transparency due to extremely complex logic. Some examples of model simplification (Law & Mccomas, 1997):

- replacement of a great number of elements and objects by one (e.g. production cell consisting of several machines can be replaced by one object)
- borders of model, creating sub-models
- avoiding workers
- avoiding transportation
- avoiding very occasional events
- avoiding shifts

Describing workers' behaviour or transportation is very complex and difficult and, in some cases, it can be simplified or even omitted.

The strategy *"20/80"* is often used in the meaning that 20% of the model size (number of elements, inputs, outputs, logic) can achieve almost 80% of the model accuracy. It can reduce time for model preparation and costs, as shown in Figure 35. There are three basic regularities in the size models:

- Model size based on time required to its construction
- Accuracy and model size
- Accuracy of model and cost for its construction

The simplification of simulation models is also important because, in general, analyses made by DES are time-consuming and expensive. Those are the general disadvantages as well as the price of simulation software and the need for simulation specialists.

One of the most critical points of the use of simulation is that it can lead to adding too much detail to the model and spending too much time in model development. Therefore, the original goals and project time lines are side-tracked. There is no guarantee that the simulation model gives accurately data and results, even if the model passed verification successfully.

Also, the problem could be associated with some type of misunderstanding of simulation results – a wrong interpretation of results can lead to premature conclusions and/or bad decisions.

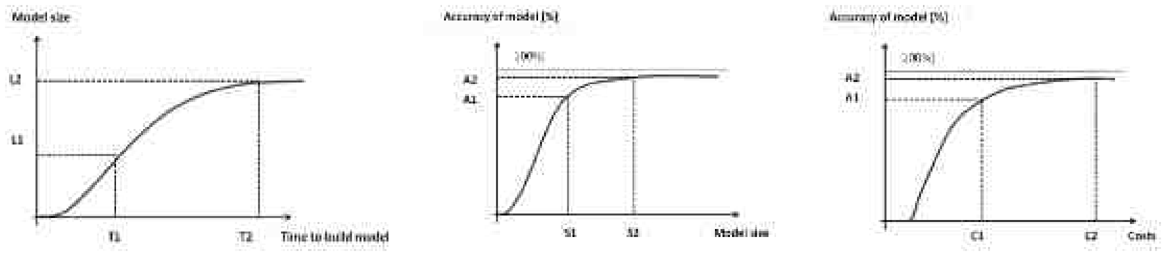


Figure 35 - Model size regularities and effects of simplification

c) <u>Simulation in the production system design</u>

DES is a powerful tool for solving tasks from the area of Facility design, planning and logistics. Simulation model can replace a real production system and, running experiments with different parameters, it is possible to find optimum system configurations for the given aspects.

Simulation models are usually used for setting the most common tasks (see Table 8) that are very helpful in the Facility design (Law & Kelton, 1991).

There are many studies and works concerning the use of DES models for layout designing to solve many tasks: system analysis incorporating **stochastic behaviour** and uncertainty (Kulturel-Konak, Smith, & Norman, 2004); **development of** a series of improved **layouts** that have been generated using traditional facility layout routines or algorithms (Altinkilinc, 2004); generation of

random flow volumes to be subsequently supplied with traditional facility layout routines (Gupta, 1986); experimentation with different layout **configurations** in terms of **operational parameters**, such as utilization, flow-time and buffer sizes (Cho, Moon, & Yun, 1996)(Hamamoto, 1999)(Adusumilli & Wright, 2004); selection of the **best strategy for the operation** of the facility (Pegden, Shannon, Sadowski, & Corp, 1995) or designing **group technology** and flexible manufacturing systems (Ranky & Morales, 2003).

Table 8 – Frequent tasks for simulation

Tasks	**Reasons**	**Comments**
Optimal usage of resources (machines, buffers, human resources, etc.) based on states	Better use of the workforce, equipment and services. Determination of adequate number of resources	Resource states can be busy (working), idle, breakdown, blocked, etc.)
Establishing storages and buffer sizes	Elimination of unnecessary occupied areas	
Design of transportation system and resources supplying, optimal transportation batches	Reduction of material handling activities Selection optimal kind of transportation system	Measuring total travel time and distances, Material handling activities (transportation, storing, packing, unpacking, etc.)
Logic control, planning of optimal material flows	Congestion reduction	
Production scheduling and optimal production batches	Reduction on delays and manufacturing time	
System performance analysis	Improving global system performance	throughput work-in-process time-in-system
Policy of production controlling	Improvement on control and supervision Moral and workers satisfaction increase	
Identification of bottle – necks in system	Possibilities of increasing production capacity	
Analysis of influences changes in production (product volume or mix)	Better adjustment to changing conditions	Testing layouts for flexibility, robustness
Evaluation and comparison of layout concepts and alternatives	Improving layouts arrangement	
Influences of uncertainties – (e.g. breakdowns machines, scraps, stochastic times etc.)	Better adjustment to changing conditions Quality control	Using of stochastic functions
Visualisation of processes and system (using animation and 3D graphic)	Presentation of results, Team's common understanding, Training and educational issues	Project results can be easily understood by managers, supervisors, engineers and workers, employers from different specialization

d) <u>Simulation tools</u>

The current marketplace is plenty of programs for discrete event simulation. It is very hard to compare them according to functions and features. List of the most popular DES applications is established based on an extensive comparison of intensity or level of presence on: Winter Simulation Conference scientific publications, document database oriented sites, social networks, internet and a selected set of sources (e.g. scientific surveys, lists and homepages) (Dias, Pereira & Rodrigues, 2007)(Dias, Pereira, Vik, & Oliveira, 2011). A list of some DES tools is presented in Table 9.

Table 9 – Simulation tools

• ANYLOGIC (www.xjtek.com, 2011)	• PROMODEL(www.promodel.com, 2011)
• ARENA (www.arenasimulation.com, 2011)	• SIMIO (www.simio.com, 2011)
• AUTOMOD (www.appliedmaterials.com, 2011)	• SIMUL8 (www.simul8.com, 2011)
• EXTENDSim (www.extendsim.com, 2011)	• SimPro (www.sdz.de, 2011)
• FLEXSIM (www.flexsim.com, 2011)	• QUEST (www.delmia.com, 2011)
• PLANT Simulation (www.plm.automation.siemens.com, 2011)	• WITNESS (www.lanner.com, 2011)

e) <u>Automatic generation of simulation models</u>

AGSM (*Automatic Generation of Simulation Models* or "*Data-Driven Model Generation*") means that simulation models are automatically generated by algorithms instead of being built by users. Mathewson (1984) defined AGSM as *"…to translate the logic of a model described in a relatively general symbolism into the code of a simulation language…"*

Historically, the idea of AGSM is almost as old as simulation itself. The first studies and prototypes of it date from 1965. Ginsberg & Markowitz (1967) used AGSM for a job shop simulation. MISDESM (Davies, 1976), DRAFT (Mathewson, 1984), SGOS solutions were used in the simulation with the generator of operational systems (Yuan, Dogan, & Viegelahn, 1993). Christenson & Dogan (1995) introduced the generator for kanban-controlled shops, Murray & Sheppard (1988) designed a KBMC for knowledge-based model construction, Schroer & Tseng (1989) used AMPS for manufacturing system simulation and Erraguntla et al. (1994) developed PROSIM for business processes.

Currently, with the increasing use of digital factory systems (see Section 2.4.3), AGSM can be used in this type of systems as described in the following works from automotive industry (Wurdig & Wacker, 2008)(Rooks, 2009)(Reinhart, 2002).

Another conceptual framework is described in the work of Lee et al. (2000). Objectives of this work are to scheme a framework to generate a WITNESS simulation model automatically from graph-based process plan and resources configuration. The solution is described on a case study – prototype of flexible manufacturing system.

Smutkupt & Wimonkasame (2009) presented in their work a concept of the integration of two modules (plant layout design module and plant layout simulation module, developed by VBA as Graphic User Interface, GUI). Through them, it is possible to optimise a layout using CRAFT method (minimizing material flows). Then, Arena simulation model can provide more information about the system, as the total time for production, utilisation, etc.

Mujber et al. (2005) presented a complex solution of integration – AGSM (WITNESS) based on data from MS Excel sheets containing information about machines, parts, processes and production schedules; and computer-generated virtual reality environment of simulation model (3D world) by Superscape VRT. The simulation can run as a background process, controlled by a user form (GUI) externally.

Son & Wysk (2001) presents a structure of AGSM (Arena) based on data from database and states from resources to basic system control. The same author also describes the use of neutral component libraries to generate simulation models. He tried to develop neutral libraries of simulation of components and model templates (containing detailed information of all the commonly used simulation components as queues, machines, etc.). These neutral libraries should be used as data sources ("*neutral model description*") for model builder that can generate simulation models for a specific commercial package. In this work Arena and ProModel builders are presented.

There are several approaches of AGSM. **Parametric** models are based on libraries blocks that are loaded into the model and configured by parameters. **Structural** models are generated based on a system structure, e.g. based on facility/factory layout (CAD systems). **Hybrid-knowledge** based models are a combination of AI methods (neuronal nets, expert system) and parametric/structural approaches (Bergmann & Strassburger, 2010).

AGSM can be a very helpful approach to improving Facility planning and design, mainly by reducing time, avoiding errors or the independent use on user´s simulation knowledge.

There are significant challenges in this area concerning data, control logic, feedback and validation of models. Models are based on input data. There can be problems with **incomplete data** set, mainly in the early phases of the layout planning when the system is in a low level of details. Simulation models used for an operation phase (describing complete production system) can use data from e.g. ERP system. In general, the hardest problem in the development of simulation models is **control logic** description. Some specific control strategies are not simply generated and they must be described by the user. AGSM should involve changes in the model and feedbacks: as structure, data or required result changes as well as required level of details in the simulation model. The problematic part of AGSM is also the **validation** of models: comparing simulation model with a real system.

2.3.3 CAD and graphic systems tools

Computer Aided Design (or CAD) is the general name for software applications used for design. By using CAD, drawings are more easily created, stored, discussed, etc. Technical and engineering drawings involve much other information, not only shapes data. For example construction drawings contain other information, for instance, used materials, technological processes, dimensions, tolerances, etc. CAD systems, as graphic applications, used graphical objects as 2D, 3D curves, surfaces, solids objects. In the case of industrial area, vector-based graphics are used whereas in the area of art, it is used raster-based graphics.

The use of CAD systems as general tools has been spread in many areas of engineering – automotive, electro-technical, architectural (Computer-aided architectural design, CAAD), shipbuilding, aerospace design, etc. It has also been spread in many different applications, from product construction design (Computer-Aided Design, CAD, Computer-aided design and drafting, CADD), assembly and technological design (Computer-aided engineering, CAE, Finite element analysis , FEA, Computer-aided manufacturing, CAM, Computer numerical control, CNC) to **Facility planning and design** (Computer-aided process planning, CAPP, Product lifecycle management (PLM), etc.

There are a great number of CAD systems in the market that can be used for the purpose of Facility planning and design. The most common widespread and well-known are:

- ArchiCAD (www.graphisoft.com, 2011)
- AutoCAD (Inventor) (www.autodesk.com/autocad, 2011)
- CATIA (www.3ds.com, 2011)
- FreeCAD (www.freecad.com, 2011)
- MicroStation (www.bentley.com, 2011)
- Pro/ENGINEER(Parametric Technology Corporation) (www.ptc.com, 2011)
- Solid Edge (www.solidedge.co.uk, 2011)
- VariCAD (www.varicad.com, 2011)

CAD systems are graphic tools and systems for **presentation**. Presentation can be as simple 2D graphic created, e.g. in MS PowerPoint (www.office.microsoft.com/en-us/powerpoint/, 2011), or based on 3D graphic and animation. Free flights are often used in designed 3D layouts in the CAD systems. Many simulation tools support **3D animation** as part of the simulation to easily understand the model. There are graphical features as textures, lights, shadows, transparency colours, high polygons models, realistic animation, etc. that help to achieve a model that is similar to the real system. This can be done in the graphical systems as:

- 3D studio max (www.usa.autodesk.com/3ds-max/, 2011)
- Maya(www.usa.autodesk.com/maya, 2011)
- Cinema 4D (www.maxon.net, 2011)
- Blender (www.blender.org, 2011)

It is possible to use systems of **virtual reality** to see these models. 3D animation and virtual reality can be used in the design of workplace, especially in ergonomics analysis.

2.4 Software tools integration

The previously mentioned tools and systems are often used only for a specific task within facility layout planning and design. They can be integrated in order to achieve a greater volume of possible functions, mainly in the area of automation of some procedures. Trough software tool integration, it is possible to get new functions and properties as data sharing in the project team, saving time by automation of some procedures, avoiding error in the design, etc.

Software tools introduced in this section do support some level of integration – either through simple data connection between two systems, or through some extension of CAD systems or even through full integration of several tools as in systems of Digital factories.

Some of the mentioned tools are professional and applied in many enterprises; some of them are theoretical or single-purpose application (prototypes).

2.4.1 Integration strategy

There are several ways to integrate applications: Object Oriented, Application or Data Based Integration (Mecklenburg, 2001).

Object Oriented Integration

Each object carries all data and methods with itself and it tells each application how to interpret its behaviour and information. This approach requires a high degree of standardization across many applications – and this is an almost impossible task.

Example: a production machine can be represented as a drawing in the CAD system. When the same object is used in DES, information as cycle time, meantime between fails, mean-time to repair, its simple graphic representation (2D icon), etc. are loaded from this object. This object can also provide electric or pneumatic schema for appropriate technical systems, etc.

Application Integration

It is based on those objects (functions) and features of one application that are available in another application. It is possible to achieve it by two ways: **container systems** or **system enhancement**. The first approach can be the following: factory layout is designed in the CAD system that is also a container to run a simulation from external simulation programme/engine. So objects in CAD are linked to the simulation objects, and in the CAD system is built a simulation model. On the other hand, DES system can involve improved graphics and drawing functions to achieve the same plant layout as it is designed in the CAD system. This is an example of a system enhancements kind of integration.

In both cases, there is no need to exchange data or do some additional work, but it is possible to merge a pair of applications. More applications mean enormous implemented functions, etc. and it is also a very complex task from the IT point of view.

Data Based Integration

All data and information are saved in one database and appropriate data is loaded by applications. The database manages and synchronizes data between the applications. This approach seems to be the most promising concept due to the process of feedbacks, the possibility of integration of more applications, information visibility and changes without direct access to the appropriate application, etc.

2.4.2 Examples of integrated tools

a) MatFlow

MatFlow is a material flow planning system, fully-integrated in AutoCAD. It involves several functions for layout design. Firstly, it supports the optimization of facility layout based on material flows values. It can be processed by two algorithms: *"Maximum method"* and *"Exchange method"*. Maximum method is a construction algorithm based on Graph Theory; exchange method is based on exchanging objects position in the layout and calculating material flow sums to achieve minimum. Optimization can also be done manually, whenever the position of facilities in the layout is changed - the layout is evaluated by a control number to easily see possible improvements.

The second significant function is the possibility to create simulation models (WITNESS) through the generation of "*lst*" file. This file contains a list of necessary elements (machines, parts), part routings, machine cycle times, setup times and initial buffer size. Based on the simulation model results, updated buffer sizes can be implemented, bottlenecks can be avoided, etc. The design layout can be continued with the use of WITNESS extension modules – Virtual Reality or Optimiser.

The disadvantages of this solution are missing automatic feedback from simulation to layout, a simple part routing (deterministic), no support of assembly and transportation operations, a simple source of inputting data (Excel sheet) and also the fact that the language support is only in German (Markt & Mayer, 1997).

In Figure 37, there are snapshots from MatFlow application – 2D and 3D layout with material flows, generated "*lst*" file and WITNESS simulation model.

b) <u>FastDesign</u>

It is an extension for AutoCAD, consisting of two applications – *FastPlan* and *FastDesign*. FastPlan is the manager interface for project database (MS Access), including also analysis functions as *Pareto analysis* (see Section 2.1.1), *Cluster analysis* (see Section 2.1.2). Data from the database are inputs that are used for the automatic generation of layout pattern (as straight production line or U-Shape). There are a lot of functions prepared for layout design: such as a large 3D drawing library of equipment and machines, a support to design buildings (walls, doors, steps, pillars etc.) and also displaying material flows for specific product or traffic intensity.

The disadvantages are: missing feedbacks from layout to database, the design is only based on deterministic values of material flows, no integration to simulation and the language support is only in German (www.projecteam.de, 2011).

In Figure 37, there are some snapshots from FastDesign – layout with material flows, 3D library, database structure and cluster analysis form.

c) <u>ProPlanner</u>

ProPlanner is a complex set of tools for facility layout design. In Figure 38, there is a set of snapshots from modules embedded in ProPlanner.

FlowPlanner module is integrated with AutoCAD; it has two main functions – displaying material flows and the calculation of distances, cost and time of flows. Material flows are automatically generated as lines with different colours and width. These flows can be straight-line, aisle-based flow or user defined flows.

Module **Relationship Layout Planning** works with relationship chart; relationships are automatically generated in the layout as coloured lines connecting each pair of facilities to easily see the required adjacency. It is based on Richard Muther's *Systemic Layout Planning* techniques (Muther et al., 1973).

Workplace Planner is a professional tool that supports a design of workplaces – displaying operator's movements, times and simple ergonomic analysis. Data are loaded from external files in "*csv*" format, involving a list of parts, a list of stations, part routings, etc.

ProBalance is a module to design and optimise production lines by tact balancing. It is provided by two methods - minimizing the number of work stations based on the given tact time or minimizing the tact time for the operations.

Other tools are: PFEP Logistics Database, Assembly Planner and ProTime Estimation (www.proplanner.com, 2011).

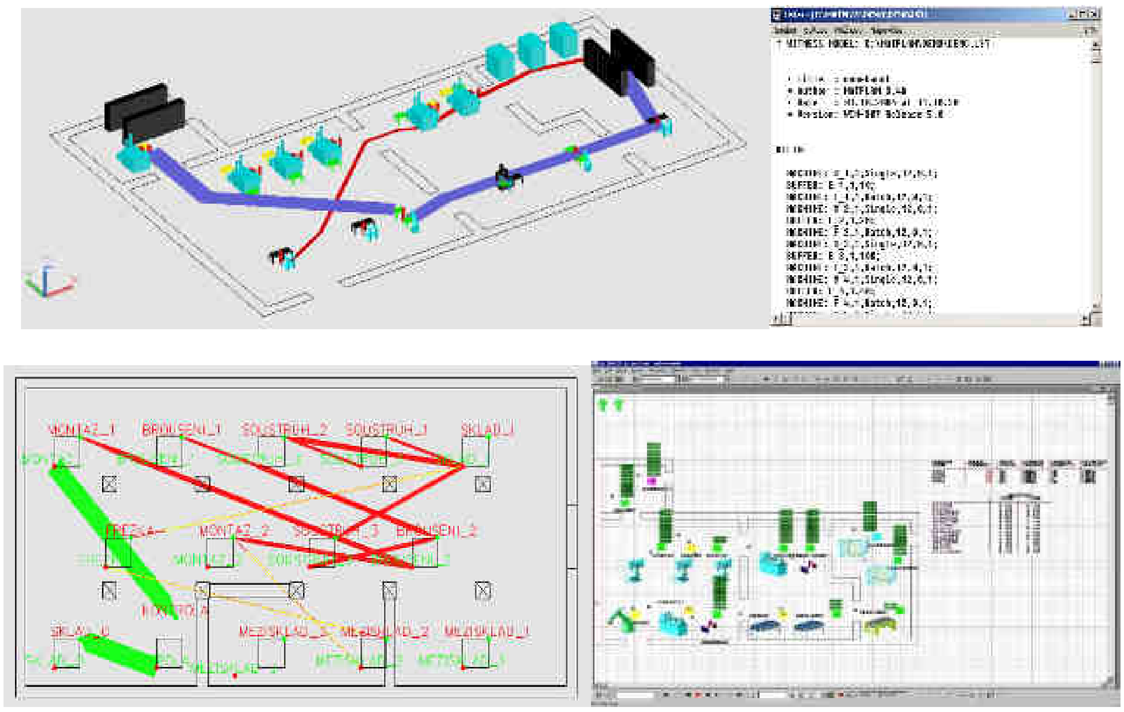

```
* WITNESS MODEL: [illegible]

* [illegible]
* Author : MATFLOW [illegible]
* Date   : [illegible]
* Version: [illegible]

[illegible]

MACHINE: [illegible],Single,[illegible];
BUFFER: [illegible];
MACHINE: [illegible],Batch,[illegible];
MACHINE: [illegible],Single,[illegible];
BUFFER: [illegible];
MACHINE: [illegible],Batch,12,[illegible];
MACHINE: [illegible],Single,[illegible];
BUFFER: [illegible],100;
MACHINE: [illegible],Batch,[illegible];
MACHINE: [illegible],Single,12,[illegible];
BUFFER: [illegible];
MACHINE: [illegible],Batch,12,[illegible];
```

Figure 36 – Snapshots from MatFlow

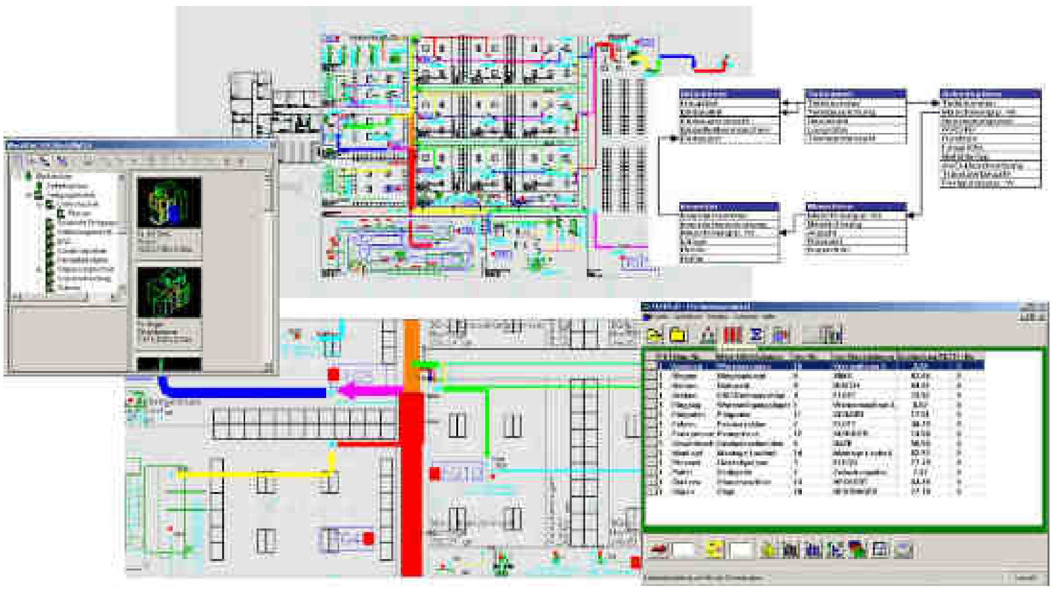


Figure 37 – Snapshots from FastDesign

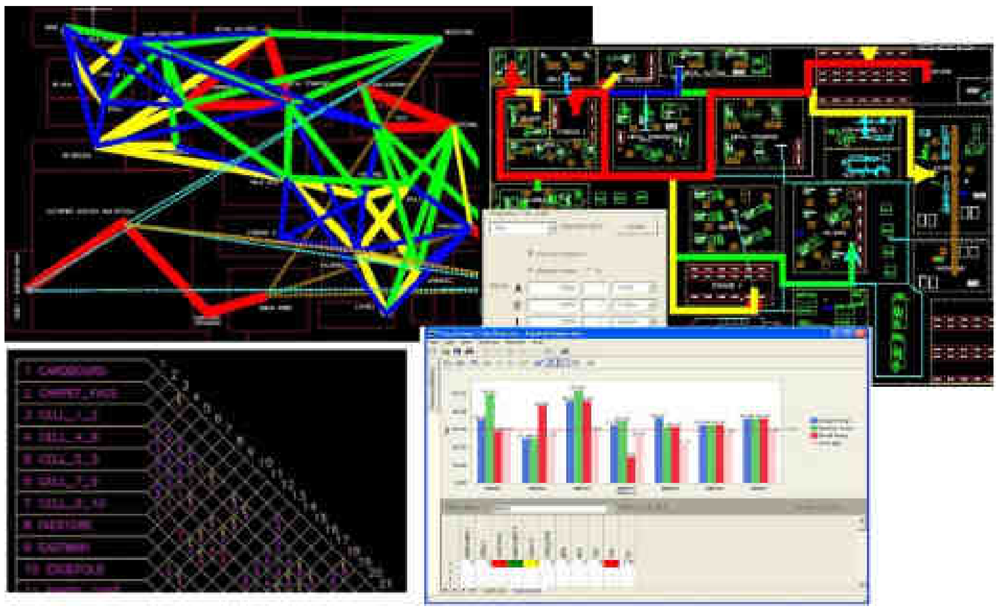


Figure 38 – Snapshots from ProPlanner

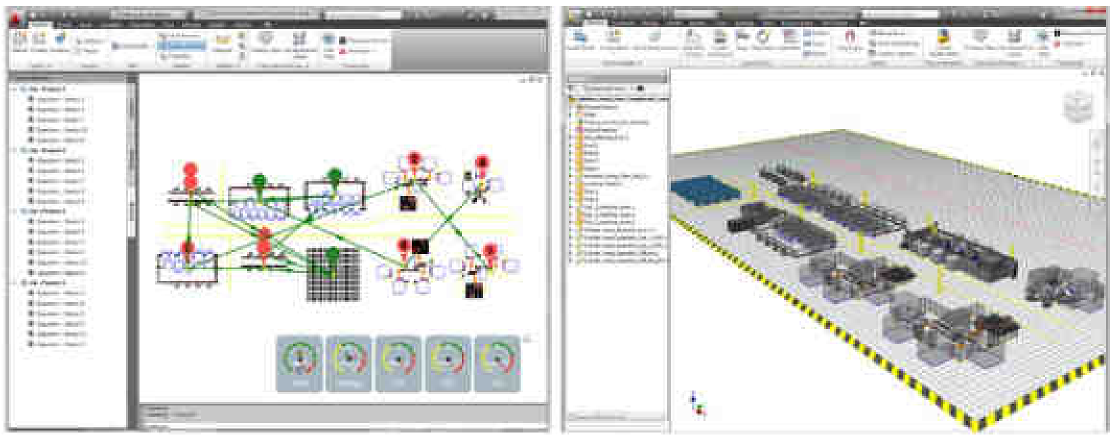


Figure 39 – Snapshots from AutoCAD Factory Design

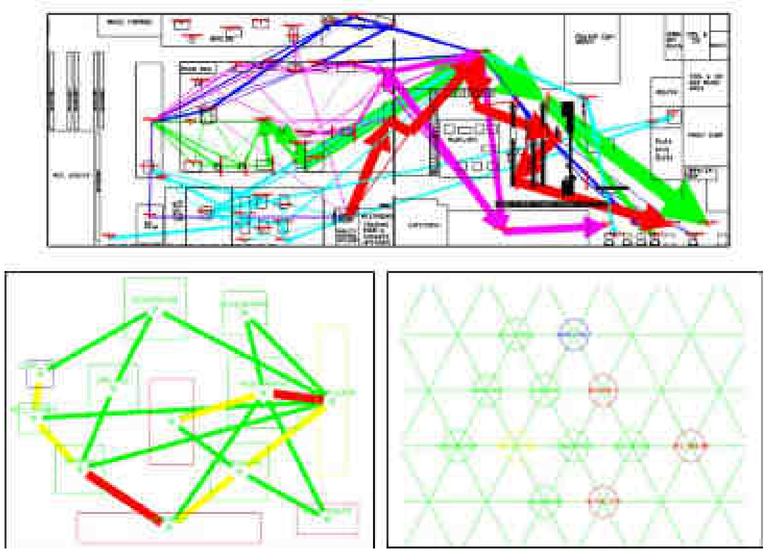


Figure 40 – Snapshots from FastDesign - Product flow diagram, Activity Relationship Diagram and Spannig Tree Diagram

d) AutoCAD Factory Design Suite

This is quite a new tool (first release in 2010) that enables design factories to involve several applications as AutoCAD® Architecture, Autodesk® Factory Design Utilities, and AutoCAD® Plant 3D and others. In general, it is possible to suggest material flows and analyse their efficiency, build a factory layout by a part from library, tools for creating 3D layout (automatic converting 2D drawings into 3D layouts, 3D virtual walk-throughs), analyse clashes and space constraints, simulation of material movement, etc.(www.autodesk.com, 2011).

e) FactoryPLAN, FactoryOpt, FactoryFlow package

FactoryFlow is an extension of AutoCAD system. It is written in AutoLisp and C programming languages. This application is focused on the material flows displaying. There are also cost and distance algorithm which extract the current activity area positions and distance of flows and create a report of travel costs, distances and intensities. The material flow is distinguished by colours, layers, thickness and arrow (direction). Specific kinds of flows are a composite that shows the total intensity between two areas (departments).

FactoryPlan is an application that designs layout based on relationship diagrams and on *Systematic Layout Planning.* This design starts with a relationship analysis and prepares an appropriate chart. Then, a schematic layout is generated.

FactoryOpt is an optimization tool, generating also layout´s facility arrangements based on a spanning tree algorithm. The layout can be a node diagram or a simple block based layout.

FactorySim is a tool that visualise material flows in the FactoryCAD and also prepare data (**SDX format**) for simulation tools (Sly, 1993, 1997).

These mentioned tools were joined in a complete software package "VisFactory". Currently, it is a part of the digital factory "Tecnomatix" by Siemens (www.siemens.com/tecnomatix, 2011).

f) SDX

SDX (Simulation Data Exchange) is a data format that can be an input for an automatic generation of DES models such as AutoMOD, WITNESS, Taylor ED, Arena, ProModel, etc.

Data relevant for a simulation (e.g. type of objects, cycle times, speeds, time to fail, time to repair) can be involved in *Intelligent CAD objects.* It allows generation of simulation model and 3D animation directly and automatically from CAD drawings. Data of inputting parts, volumes, routings, setup times, assembly tree structures, etc. are uploaded to an external database. This type of information is added to the SDX. Simulation model can be then automatically generated in the supported simulation tool by its inner translator (Moorthy, 2002; Sly & Moorthy, 2001). In Figure 41, there is a schema of SDX architecture.

The advantage of this integrated approach is the possibility of using many DES tools as a universal format. The disadvantages are: missing feedback information flow from DES model back to layout, the need to use "intelligent CAD objects" involving some data, splitting data sources – while some data are saved in the smart object in CAD, other data are saved in the external database.

Practical use of SDX in FactoryCAD is described in the work of Mecklenburg (2001) as well as problems as data synchronising (data-exchange, feedbacks), data preparation and building simulation model in AutoMOD and WITNESS. Currently, the used drawing blocks of production equipment must be changed into "intelligent CAD block" holding data; some equipment is not supported.

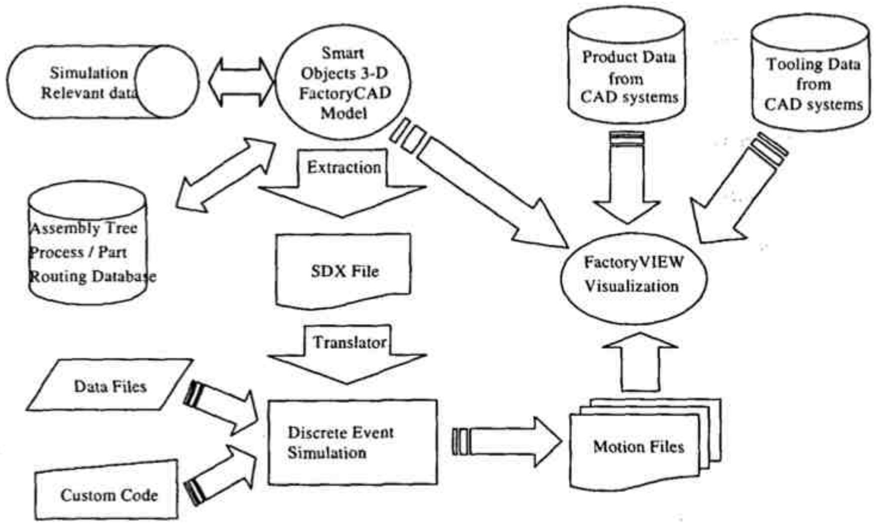

Simulation Relevant data
Smart Objects 3-D FactoryCAD Model
Product Data from CAD systems
Tooling Data from CAD systems
Extraction
Assembly Tree Process / Part Routing Database
SDX File
FactoryVIEW Visualization
Translator
Data Files
Discrete Event Simulation
Motion Files
Custom Code

Figure 41 – Simulation Data Exchange Architecture (adapted from (Moorthy, 2002))

g) Theoretical or single-purpose application

A large number of works focusing on some type of integration were written. The following works and papers were chosen:

Integration possibilities of Arena with some external tools (VISIO, MS Access, MS Excel) are introduced in the work of Seppanen and Marvin (2000) where the example Arena-Excel connection is described, mainly reading/writing data from/to simulation model, the control of experiments and building *Industrial strength simulation models* by VBA macros.

Mertins & Rabe (1995) described integration of MOSYS simulation tool and GDS CAD system as "mutual communication module" that enables on-line use of data-file and connection of two software systems.

Vilarinho & Guimaraes (2003) presented a conceptual and theoretical framework for decision support system in the area of FLP. It was formed a modular system covering the whole process of system design where optimization algorithms are used in each phase.

Planning stockyard processes based on simulation and on the CAD system were introduced by Marasini (2002) as a prototype "SimStock". He tries to integrate production information and space requirements to optimise stockyard processes, mainly effective allocation of products, effective routing of vehicles and the use of forecasting. AutoCAD is used for designing spatial layout of storage area, roads and aisle. This information is stored in the database by VBA macros.

2.4.3 Digital factories

Digital factory (DF) is a complex application structure that creates a virtual mirror of a real/planned system. It shows the production processes in the virtual environment. Figure 42 shows a principle of DF. The general aim is the data integration of three areas of the whole product lifecycle called *Product Lifecycle Management* (PLM). These entire data are involved in *Production Data Management* (PDM). Three basic areas of PLM are **CAD/CAM** systems covering product design, *Manufacturing Process Management* (**MPM**) controlling manufacturing and **ERP** systems for planning and scheduling production. System of the DF covers a data-gap between CAD and ERP systems.

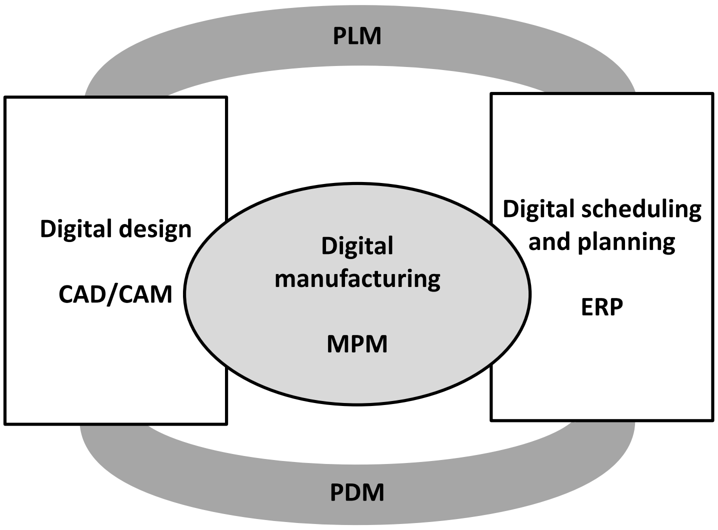


Figure 42 – PLM software

Definition of digital factories

The definition of the DF system is difficult because of the existence of several points of view and, therefore, a general definition hasn´t been accepted yet. Chen's analysis shows more than 60 works and the conclusion is that almost every author has different perception and definition of the DF systems (Chen & Kjellberg, 2009) Even names for this type of approach are mentioned, such as "*Digital factory*", "*Digitalefabrik*", "*Digital plant*", "*Virtual enterprise*", "*e-Plant*", "*e-Factory*", etc. One reason is to make digital factory a vision, based on unlimited features of digital world;

another reason is quite a short history of DF.

- 1998-2000 – new possibilities of CAD/CAM and ERP systems, establishing to CIM theory by modern and advanced computer technique, first experiments
- 2001 – 2003 - feasible plan analysis, pilot project of the largest automotive and aircraft companies, proofs of advantages
- 2004 – 2006 – idea of concurrent engineering also used in another type of industry (ship, mining etc.)
- 2007 – present – suppliers of automotive industry are forced to used DF by their customers, decreasing prices, spreading to more companies, more solutions, appearance service providers (integration between systems)
- probable future development – because of the establishment of format standards and methods, DF may become a common approach in the PLM

Association of German Engineers (Zülch & Stowasser, 2005):

"…The digital factory is the generic term for a comprehensive network of digital models, methods, and tools – including simulation and 3-D / virtual-reality visualisation, which are integrated by a continuous data management system. Its purpose is the integrated planning, evaluation, and continuous improvement of all essential processes and resources of the factory in combination with the product…"

University Stuttgart (Westkaemper, Jendoubi, Eissele, & Ertl, 2005):

"…The digital factory can be seen as a static model that includes geometry, technical and logistical data, or an image of the factory's objects. The digital factory contains the digital information about the factory and its resources, i.e. location, media, logistics, simulation tools etc…"

On the other hand (T Kjellberg & Katz, 2005) states that

"…The digital factory is a generic digitalized model of a factory with its manufacturing system model as its core - e.g. often a **combined set of models into a whole**. *I.e. in the digital factory the information about the manufacturing system should mirror the real manufacturing system. In factory design the information (and its data representation) is evolving from the initial stage of the design to final detailing and during various stages of reconfiguration. Information on production equipment and its capability, tooling, fixtures, materials handling devices etc. is needed. The digital factory should form the information platform for manufacturing system's life cycle management…"*

Bosch (Blumenau & Kotz, 2004):

"…The basics of the digital factory are: latest methods and processes in planning, integration of software tools, embedding in the organization of Bosch and personal qualification…"

Based on the mentioned definition, DF can be taken as a simple database system that integrates different applications of production system through an animated visualization and simulation to a completely digitalised world based on the virtual reality vision. However, Nylund et al. (2008) find common characteristics in digital manufacturing, factories and enterprises:

- Integrated approach to improve product and production engineering processes and technology
- Modelling, simulation, planning and analysing of real manufacturing processes
- Collection of systems and methods supporting new technologies

Advantages and disadvantages

The general advantages can be summarised such as time and financial savings, sharing data, team cooperation and work, integration of tools and visualisation of processes. The total time spent on the project can be reduced because of a faster team communication within the DF system. DF supports work in the team composed of product designers, technologists, logistics, in production managers, etc.

The whole data are saved in the common project database, containing the actual data. It avoids having redundant data-storage or data duplicates that are often source of errors. Digital data and knowledge can be used in several phases or projects.

The concept of integration tools supports changes in the project and feedbacks. Alternatives can be validated by simulation models and it is possible to process global optimization of production system with all resources, such as equipment, space, material flows, etc. This way it is possible to achieve the desired system features as modularity, flexibility and robustness.

DF usually supports virtual reality visualisation. Tools such as ergonomics, programming of robots off-line or the use of physical prototypes (digital mock-ups) can also be integrated.

Figure 43 displays the comparison between the traditional approach of production preparation and the use of DF system. Costs are increasing in the preparation phases but the stabilisation of the production system (aim) is achieved in a shorter time, which means cost savings in later phases, e.g. in system fine-tuning. The most expensive phase is making changes in the real system or in the concept in later phases of the planning process

Using DF, it is possible to validate the system concept before building a real system (Leeder et al., 2006).

On the other hand, there are disadvantages, such as the high **price** of the software and the requirements for **staff** and **training**. There is a great pressure for having the appropriate level of **data** for using DF and these data must be held actual. The problematic part of the DF system use is its **implementation**; the replacement of the currently used software tools can be painful, especially if the software has been used for a long time and if there is a high level of knowledge in the departments. An example can be a replacement of the widespread simulation tools (WITNESS, Arena or AutoMod) by tools included in the DF packages (Plant Simulation or Quest). Problems can arise when transferring data from the currently used systems to DF and data format change. The combination of many related tools can make a **complexity** of the system usage. The mentioned issues usually bring some extra costs.

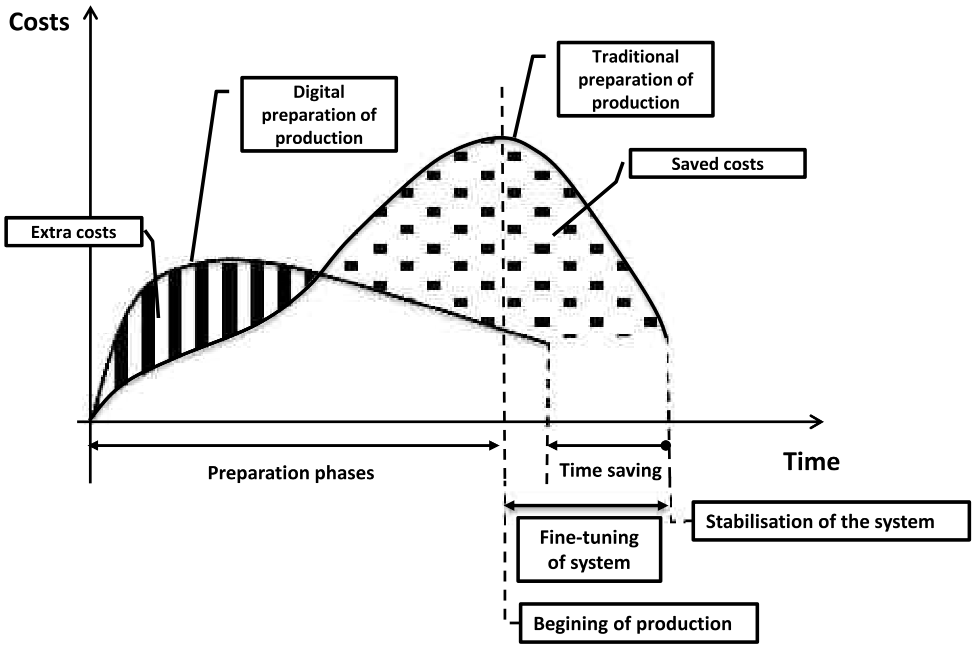


Figure 43 – Product lifecycle and the use of Digital factories

Measured results

According to Curran et al. (2008), Digital Factory enables to achieve the following resource **savings:**

- Area savings by layout optimization about 25 %
- Cost savings by a better use of resources about 30 %
- Cost savings by material flows optimization about 35 %
- Reduction in number of machines, tools, workplaces about 40 %
- Total cost reduction about 13 %,
- Production volumes growth about 15 %,
- Time to market reduction about 30 %.

State-of-the-art of digital factories system

Currently, there are only two commercial systems of DFs: **Process Engineer** (or **Delmia**) and **Siemens PLM** (or **Tecnomatix**). They have been for several years on the market and many upgraded versions have been released but they are not widely used yet; only in international companies (automotive, aircraft, shipyards industry) with high demands on the manufacturing.

In the work of Chen (2009), there is a critical comparison between both systems – functionality, integration, used tools and also a different meaning of a term of DF.

These systems are very complex and huge; a brief introduction of these commercial tools is presented below.

Delmia Digital Factory

This system, also called *"Process Engineer"*, is developed by the French company *Dassault Systémes*. This company also creates a system called *CATIA,* focused on construction and technology.

The core of the Delmia is The PPP (Product-Process-Resource) that connects the whole content of DF with all logical relationships. It supports simultaneous work ("*concurrent engineering*") of several team users, as constructers, technologists, etc. and it also enables access to current data immediately. So e.g. technologist can influence construction design based on a type of production system. Delmia consists of several **modules** (Ergonomics, Robotics, NC programming, simulation, process planning etc.), as it is shown in Figure 44.

Quest can be used completely independent on the Delmia system when a simulation model is made manually or as a part of DF. In this case, prepared data are loaded into Quest and a model is automatically generated, experiments are run and results are implemented back to DF. Changes in DF are implemented in the simulation models by its updates or by the generation of new models (Leeder et al., 2007)(www.delmia.com, 2011) (Bzymek, Nunez, Li, & Powers, 2008).

Tecnomatix digital factory

Tecnomatix is a product of the company ***Siemens PLM Software***. In 2007, "*Siemens Automation and Drives group*" that bought a former developer – company *UGS*. It is digital manufacturing software within the "e-Factory" domain.

Similarly as Delmia, Tecnomatix (or Siemens PLM) covers the whole product lifecycle and it consists of several modules. It involves a simulation tool ***Plant Simulation*** with functions as Bottleneck analyser, Optimization by genetic algorithms, Sankey chart or Gantt chart. For the layout design, it is included ***FactoryCAD***, ***FactoryFlow and FactoryPlan*** tools using a smart object (see more in Section 2.4.2). Tecnomatix involves also the ergonomic tool "*Jack and Jill*" (www.siemens.com/tecnomatix, 2011) (Worn, Frey, & Keitel, 2002).

These two DF systems are very similar in functionalities as it is shown in the schemas of the included applications in Figure 44 and in Figure 45.

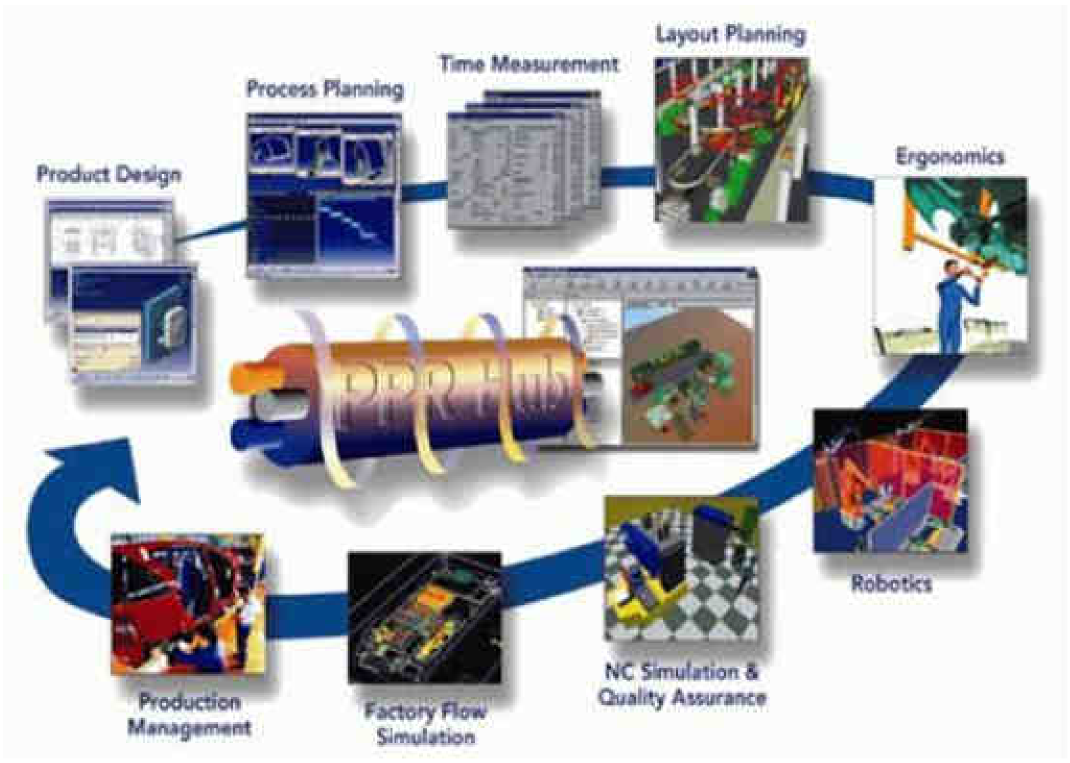
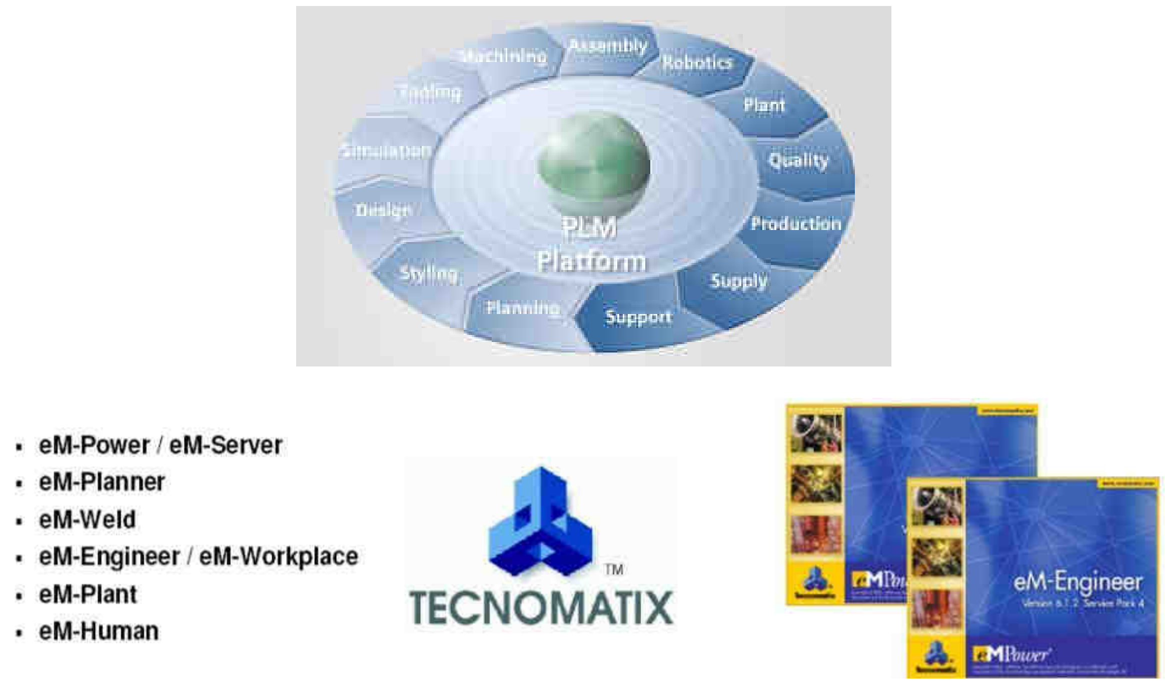


Figure 44 – Delmia – PPR Hubs



- eM-Power / eM-Server
- eM-Planner
- eM-Weld
- eM-Engineer / eM-Workplace
- eM-Plant
- eM-Human

Figure 45 – Tecnomatix DF

Research projects in the DF area

There are also several research projects as *Nexus* (Lucke, Constantinescu, & Westkämper, 2009), DiFac (www.itia.cnr.it, 2011) or *Open Digital Factory* (ODF) (www.sim-serv.com, 2011). In the work of Masurat (2009), the principle of ODF is described as integration of tools within open structure. It means that it allows docking other tools or the implementation of company-specific application at any time by the definition of standards and interfaces. By this approach, it can be possible to integrate the current used software without any extra cost with the purchase of licences or user training.

2.5 Summary

In this chapter, basics from "*Production Systems Design*" and "*Facility Layout Design*" were introduced. Methods for *Facility Layout Problem* as algorithmic methods (*CRAFT*, *CORELAP* etc.) and procedural methods (*Systematic Layout Planning*) were described. Industrial engineers use many commercial software tools for solving tasks from production systems design. These software tools can be integrated together to widen functions and increase the user's work performance, as described in the following chapter.

3 Integration Concepts and Challenges

This chapter describes the reasons for this work based on a state-of-the-art on production systems design, where most relevant challenges are discussed and the overall concept of IDS (Integrated Design of Systems) is presented.

The database concept, the simulation concept and the CAD concept were designed based on the challenges identified. Also, the expected advantages and disadvantages are discussed.

3.1 Challenges

An **impulse** for this work was the fact that there is a lack of a cost-effective software solution that supports the integration of the applications used in facility design.

In this context, a cost-effective solution is the one which returns back the investment (capital and time) through higher performance systems design in the appropriate time. That is supported by software tools analyses referred in the previous chapter.

The most significant and indivisible set of features to achieve are: **Integration**, **Easy** and **Rapid** to use, **Broadband** application field, **Multi-level** detail design and Affordable Tool **Price**.

None of the current tools offer all of these features together.

3.1.1 Need for integration

In the facility design process, CAD systems are usually used for layout design (drawing); simulation, as explained above, is a powerful tool to validate this layout. Thus, it is possible to get information about buffer sizes, bottle-necks, etc. It is very convenient to design facilities layout and material handling systems simultaneously. Information from simulation is then implemented in the layout. According to Grajo (1995), *layout optimization* and *simulation* are two tasks that are fundamental to facility planning. Burgess et al. (1993) proclaimed that simulation is the only methodology robust enough to the systematic examination of key variables of factory performance. Simulation methodology enables the representation of many attributes of real life problems that are difficult to consider in analytical models for the layout optimization problems (Tam & Li, 1991)(Tang & Abdel-Malek, 1996)(Pandey et al., 2000)(Castillo & Peters, 2002).

With no integration tool available, the effective interaction between CAD and simulation systems is difficult and time consuming.

A very common practice as far as facility layout design is concerned would involve the following steps: a 2D/3D facility layout is designed based on given tasks in a CAD system. This task is usually performed in the *Plant Layout department*. The main effort of this department is to achieve geometrical arrangement of necessary equipment, fit it together (no collision or interferences), use space optimally, etc. For the presentation of final layout alternatives, it is possible to use virtual reality or just flight mode through 3D model factory. Then, the design process can continue at the *Simulation department*, performing detailed analysis in terms of events, time, uncertainties, etc. Simulation experts need a lot of information (process data) for it - cycle times, up-time/downtime, availability and routing, from *Process Engineer*, *Tool Designer* and *Production department*. Facility layout from the *Plant Layout department* is also useful information source. Problems could arise now, concerning data formats and ways for data exchanging (at this stage, a layout could still be a paper layout). After the complete data are collected, the simulation model can be developed, experiments run and final reports would be available. This type of approach is described in the work of Mecklenburg (2001) and is also based on the author´s experience.

It is evident that there are missing data feedbacks, common shared data storage and communication pipes within the team. It causes more time for project solving, as it is shown on the left side of Figure 47. In this thesis, CAD and simulation tools will be integrated. IDS is an acronym for **I**ntegration **D**esign of **S**ystems. IDS will support fully integrated connection between applications.

With the integration of the CAD system and DES tool, it is possible to combine *static optimization* with *dynamic optimization*. Static optimization means finding optimum placement of resources in the layout based on material flows, whereas dynamic optimization is focused on features as optimal usage of resource, transportation and production batches, buffer design. These features are based on time, relative interactions and stochastic influences. In this way, it is possible to achieve a global system optimization considering all resources in the system.

3.1.2 Need for an easy solution and a rapid system design

There is also a great pressure for rapid production systems design or re-design due to the volatile market environment and uncertainties in the systems. Due to these reasons, common tasks of facility design should be performed automatically (as much as possible). In Figure 34, there is an example of simulation project phases with estimated times, showing the most time-consuming phases eventually needing improvement.

IDS will support a rapid and visual design of production systems alternatives. The rapid design will be managed by **automation** of some phases (or functions), such as the generation of simulation models, the generation of basic layout patterns, displaying material flows in the layout, etc. IDS will contain some important system analysis and methods that will help to understand the system behaviour itself and also quantitative performance measuring that helps to select optimum alternatives.

The visual design will be supported by user interface with **input/output forms**. The results will be processed into detailed graphs and summary documentations. It also helps to make facility design simpler, without duplicating tasks and minimizing errors.

This work is not focused on developing new algorithms. It will use selected algorithms, commonly applied in the industrial environments – general purpose algorithms are key algorithms for the proposed tool. They should also imply simple integration steps and the computational time should be as short as possible.

3.1.3 Need for a generalized approach and multi-level detail design

The current systems of digital factories are mainly focused on automotive production and its specific problems but these systems can also be very helpful in other areas of industry or even in services (see Table 1).

IDS should be able to support most **designing phases** and **input data details**. It should also support an approach from general to detailed facility design - as described, for example, in the work of Hassan (Hassan & Hogg, 1991). In this study, a **hierarchic approach** in facility design is considered and the required data are mentioned.

The designed solution should also be useful for generic sized companies, from small to big companies. It should be helpful in many different types of tasks, not only for the production area.

The CAD and simulation tools selected to integrate IDS, should be widespread and well-known between common users. These tools should be suitable for solving general project tasks.

3.1.4 Need for low cost solution

There are digital factories systems that support full data integration - Tecnomatix by Siemens and Delmia by Dassault Systémes, introduced in Section 2.4.3. Although the mentioned advantages, there are also some disadvantages - complex usage and high price for software (from 50K € based on current conditions and configurations). Therefore, only the largest corporations (automotive, aircraft, shipyards, etc.) can afford it. High costs associated with personal requirements (expensive training), the need for data system maintenance, data format exchanging between external suppliers and customers, etc., represent other disadvantages of present approaches.

In IDS, there will be a particular effort to integrate the most widespread software tools in production systems design. In an optimum scenario, a company showing interest in IDS implementation would not need to buy new software licenses or training personnel. IDS implementation should be as easy as possible.

3.1.5 Application on practical examples

IDS will be tested on practical examples and compared with the traditional way of designing and solving. The purpose is to demonstrate the advantages and disadvantages of the developed software tool when applied to different examples from different application areas.

3.2 IDS concept

The following section shows the development of the IDS concept based on the challenges mentioned on the previous sections.

3.2.1 Desired integration approach

An example of the possible practical use of IDS can be related to the design of buffers in a production environment (see Figure 46). It involves two types of information: optimal capacity of buffer and also the arrangement of the buffers in the required space. The first type of information could be obtained by simulation experimentation and analysis; the second one could be obtained through layout definitions. This information must be considered during the design process.

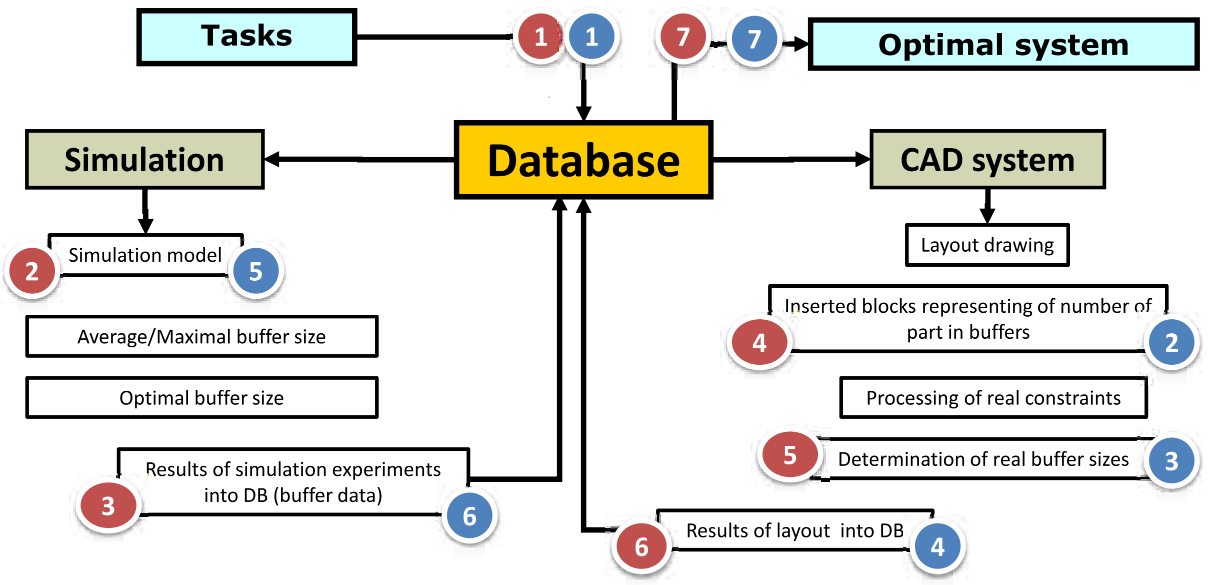

Tasks
1
1
7
7
Optimal system
Simulation
Database
CAD system
Simulation model
2
5
Average/Maximal buffer size
Optimal buffer size
Results of simulation experiments into DB (buffer data)
3
6
Layout drawing
Inserted blocks representing of number of part in buffers
4
2
Processing of real constraints
Determination of real buffer sizes
5
3
Results of layout into DB
6
4

Figure 46 – Diagram of feedbacks in system

There are two ways that could be used to achieve the aims:

1. Design the system based on the simulation model and then design the layout and facilities arrangement (red numbers in Figure 46)
2. Design the layout and facilities arrangement that are validated by simulation model (blue numbers in Figure 46)

In the first approach (red colour), the definition of production system and project tasks (1) must be defined in the common **database**. Then, the **simulation model** is built (2) and information about buffer behaviour is received by running experiments. In this case, this information includes average and maximum buffer sizes. The optimum buffer size is determined and these values are written back to DB (3). In this project phase, a question is enquired. "Is there enough space in the layout for the established buffer size?" The answer for this question can be found in the **layout**, designed in the CAD system (4). Some drawing blocks are inserted, representing stored items. Then, the available space for them is recalculated (5). The available space can be influenced by several constraints (e.g. walls, actual production equipment, aisles, etc.). This number is recorded back to DB. In that case, it happens to have less space than required. So, a new simulation experiment can be run (***new iteration cycle***) and a new question is raised: "Which consequences to the production system will be caused by lack of space for this buffer?" (2)

The second approach is very similar. It starts with a layout and then uses simulation model (blue colour). Based on the project data in DB (1), a layout involving buffers and stored parts - estimated number, is generated (2) (3) and then the layout information is saved in the DB (4). The simulation model is then generated (5) and experiments specify the buffer sizes (6). These updated values can be used in the layout.

In this suggested integrated system, there is a comprehensive set of **feedbacks** that is very important for data transfer between software tools. This should be managed fully automatically to avoid time wasting (rewriting values manually).

Both approaches have advantages and disadvantages, as it is summarised in Table 10 where both ways (paradigms) are compared (Aleisa & Lin, 2005).

The first approach optimises the system processes into the desired values and states (e.g. throughput, required storage space, resource capacities and their use, etc.) using simulation and then, the layout of the system is designed. It is convenient due to considering accurate values of material flows, involving stochastic, logic control and resource interactions. This approach can also be used in projects where random material flows are required as input to the FLP (Kulturel-Konak et al., 2004) or in cases of difficult estimation of material flows.

Pandey et al. (2000) make a model that is optimised by simulation and then adapted the results into a layout. Altinkilinc (2004) improved the system with simulation and then used a CRAFT method for layout optimization.

In the second case, the layout is designed based on deterministic flow volumes, estimated buffer sizes, etc. After that, the simulation model is built to improve operational system characteristics. Different layout alternatives are designed based on predetermined overall production strategies and technologies. Stochastic influences (flows or processes) are not significant in this design stage. Sly (1997) suggests starting with the layout scratch based on an adequate previous analysis (P-Q, relationship diagram) and then improve it with simulation modelling.

With IDS, it is possible to use both approaches because of the full applications integration, allowing feedback in both ways – in fact, both simulation models and layouts will be automatically generated.

Table 10 – Comparison of the approaches

Approach/Issues	Simulation→Layout	Layout→Simulation
General system properties	Significant stochastic behaviour Complex interactions betweeen resources, resources states (busy, waiting, blocked, etc.)	Insignificant stochastic behaviour
Type of layout	All types of layout Implementing group technology, flexible manucturing systems	Special type of layout, simulation model helps to verify it
Material flows	Accurate estimation of material flows based on stochastic resources states or control logic random flows	Deterministic volumes, predetermined values
Optimization and focuses	Bottle-necks or congestion in the system Transportation and manipulation of significant Operation sequences	Minimizing traveled distance Improving adjacency score
Facility design	Convenient for new layout design where contraints and limitation are not so strict	Convenient for re-layout (improving existing layout)
System parametres	Major operational policies/technologies are not predetermined or need to be justified prior to the layout optimization	Only minor system´s process parametres need to be adjusted
References	(Pandey et al., 2000) (Eneyo & Pannirselvam, 1998) (Altinkilinc, 2004)	(Sly, 1997)

3.2.2 Comparison between the traditional and the new approach

The main differences between the traditional (non-integrated) approach and IDS are shown in Figure 47. The traditional approach (left image) often uses tools separately or with minimum relative integration and data can be stored in several places and in different formats. On the other hand, IDS makes use of a full integration (right image). A similar idea of integration is described in other works (Chee, 2009)(Benjaafar & Sheikhzadeh, 2000)(Sly & Moorthy, 2001).

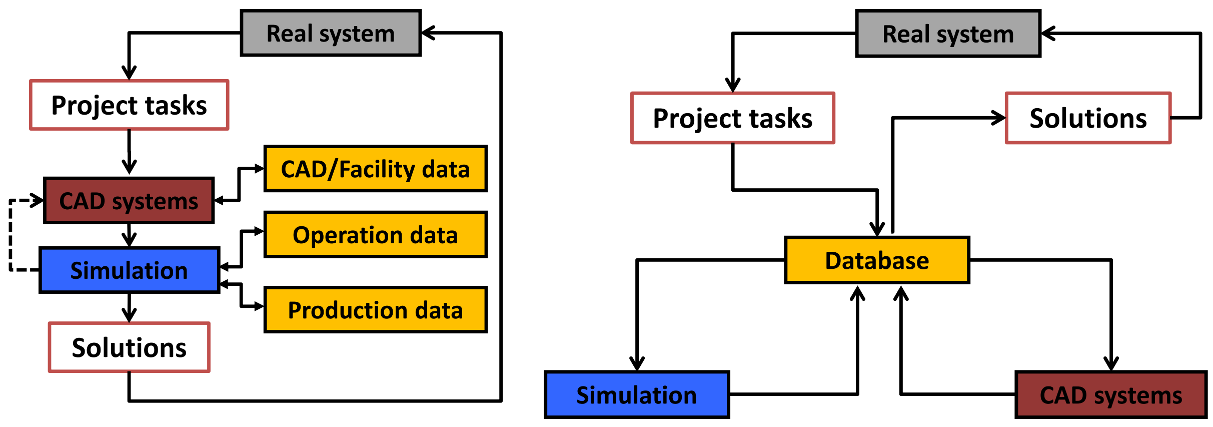


Figure 47 - Two approaches of software tools usage

Integration will be managed by one common database. It allows controlling simulation model from database and read/write the required/received data.

Advantages of the suggested integrated approach

- Bidirectional information flows, i.e. inclusive feedbacks
- One common database
- Easy data transfer and unified data format
- Fast reaction to changes during the project elaboration
- Full use of software tools and their extension by new functions

3.2.3 Database concept

A database structure design has usually the following phases: requirements analysis, conceptual design and logical design. Required features for database as well as for collected information about nature of the data, etc., are established. Then a conceptual design involves a schematic drawing of Entity Relationship Diagrams (ERD), including tables, fields and relationships between tables. In the last phase, DB is created by database language commands. In this section, the first two phases will be described. The last phase, implementation of concept, is in Section 4.2.5.

Table 11 – Table of objects used in IDS

Object	**Meaning**
Entity	active movable object; product; part properties changeable by the operations
Resources	production machine and equipment; transportation facility
Space	space for entities, storages, warehouse, buffers
Human resource	worker, labour
Crate	pallet, box or final product

a) Requirements analysis

DB has several tasks, mainly: data storage, facilitate data-bridges between simulation tool and DB and between CAD and DB, data analysis and automatic generation of simulation models through VBA macros. DB involves complete production system description with all resources and objects – the meaning of those objects is explained in Table 11.

All these tasks and functions are controlled by user Input/Output interface forms. In Figure 48, there is a simple schema of the DB required functions and relationships to external applications.

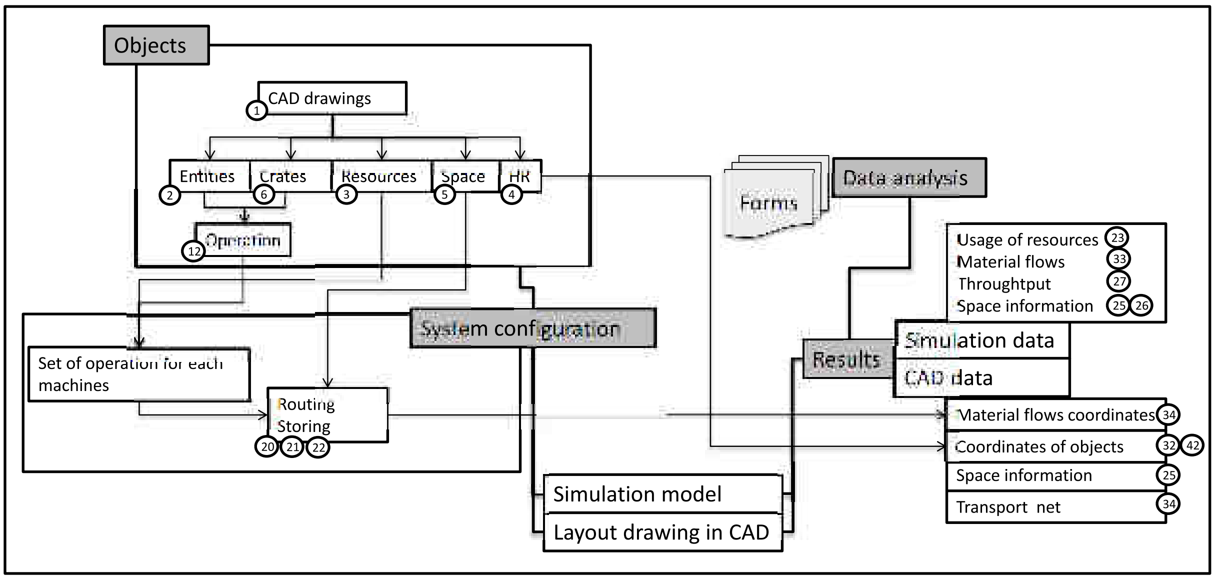


Figure 48 – Schema of DB functions and data-flows

b) Conceptual design

The design of DB must involve several types of data sets and linking them together through relations (McFedries, 2007) (Jennings, 2006).

In summary, there are:

- Lists of **objects**
- System **configuration**
- **Results** from applications and analysis

Objects are resources, entities, crates, space and human resources, as described in Table 11. In DB, objects properties, physical features, CAD drawing blocks references etc., must also be defined. **System configuration** involves information such as: a set of operations for each machine, entities routing maps and storing. Routing map is based on the set of operations for each machine and the sequence of operations that the entity must go through. Storing involves information about resources and space, especially from which space entities are loaded in the resource and to where they are to be stored. **Results** involve data from simulation experiments (usage of resources, optimum space for buffers, etc.), optimum placement of resources, data analysis and data from CAD layouts (coordinates of placed objects in layout, coordinates of transportation paths).

Database Specification

The following formulations contain the entire structure of DB. Each formulation represents data table with columns. Primary keys are signed as underlined text and in bold font; foreign keys are in italic font. The table numbers correspond to Figure 48.

DT 1 table creates a **Library of CAD blocks**; it holds information as a path to appropriate drawing files and CAD block reference, which is a unique name for the use of block in IDS.

DT 2 - **DT 6** describe **basic resources and objects** of which the system is composed: Entities, Resources, Human resource, Space and Crates. Each of these objects can be represented by a CAD drawing in the layout. Other properties can be physical information as dimensions (width, length, high) and weight. Moveable objects (entities and crates) also contain information about colour of material flows. Resources (**DT 3**) also involve "specific information" used e.g. for the speed of transporters, etc. Space (**DT 5**) is defined by the capacity and time details. Capacity is a number of crate units that is possible to store; this value is determined by a crate property stack (**DT 6**). Minimum time is the period that the entity must stay in the buffer. Each of the basic objects can also be specified by a specific type of information (**DT 7** - **DT 11**). In the case of Resource: the type of resource and the type of technology. The type of resource divides resources into production, transportation, supporting resource, etc. The type of technology specifies used technology of production machines (assembly, metal forming, drills, packing equipment etc.). Entity can be a standard product or dummy objects (a type of entity), and space can be buffer for storing, conveyor for transportation or parking space for transportation resource.

Library_CAD_drawings (**CAD block reference(P)**; drawing name and file path)	**DT 1**
Entities (**ID_entity(P)**; unique Entity name; *type of entity(F)*; simulation graphic data; CAD flows colours; *CAD block reference(F);* physical information: weight, dimensions; price)	**DT 2**
Resources (**ID_resource(P)**; unique Resource name; *type of resource(F)*; simulation graphic data; *CAD block reference(F);* physical information: weight, dimensions; price; specific information; *type of technology(F)*)	**DT 3**
Human_Resources (**ID_HR(P)**; unique HR name; *CAD block reference(F)*)	**DT 4**

Space (**<u>ID_space(P)</u>**; unique Space name; capacity; crate; *type of space (F)*; time details; physical information: dimensions; simulation graphic data; *CAD block reference(F)*)	**DT 5**
Crates (**<u>ID_crates(P)</u>**, unique Crate name; *type of crates(F)*; stack; CAD flows colours; simulation graphic data; *CAD block reference(F);* physical information: weight, dimensions; price)	**DT 6**
Type_of_Entity (**<u>type of entity(P)</u>**)	**DT 7**
Type_of_Resource (**<u>type of resource(P)</u>**)	**DT 8**
Technology (**<u>type of technology(P)</u>**)	**DT 9**
Type_of_Space (**<u>type of space(P)</u>**)	**DT 10**
Type_of_crate (**<u>type of crate(P)</u>**)	**DT 11**
Operation (**<u>ID_operation(P)</u>**, unique Operation name; *type of operation (F)*; operation time; costs; setup; blocked; unpacked; part wait)	**DT 12**
Type_of_operation (**<u>type of operation(P))</u>**	**DT 13**
Inputs *(ID_operation(F)*; *ID_entity(F)*; *ID_crates(F)*; variant; order_in_inputting_sequence; order_of_route_in_system; input_to system; number_of_entities)	**DT 14**
Outputs *(ID_operation(F)*; *ID_entity(F)*; *ID_crates(F)*; variant; order_in_outgoing_sequence; order_of_route_in_system; output_from_system; number_of_entities)	**DT 15**

Operation (**DT 12**) is defined by a unique name, an operation time, a setup time and properties of behaviour as: "blocked", "unpacking" or "part waiting". These properties are described in detail in Section A.4.2.4 on practical examples. Operation is also specified by "type of operation" (**DT 13**): production, service, storing, transportation, conveyor etc. Inputting and outgoing sequences of entities and their properties information are involved in tables Inputs (**DT 14**) and Outputs (**DT 15**). In these two tables, the items are: entity, crate, sequence of operations in the system, variant, order in inputting/outgoing sequence and a number of specific entities. "Input to"/"Output from" information system describes the limitation in the entity lifecycle in the system.

Table **DT 16** represents data on weight of entity on the crates and transportation costs.

Weight_Entities (*ID_entity (F)*; *ID_crates (F)*; weight; costs_for_transportation)	**DT 16**

Production schedules are involved in table **DT 17**. For each inputting entity, information about the arriving time and number of entities is found.

Part_Scheduling (*ID_entity (F)*; *ID_crates (F)*; order; variable; time; number_of_entities)	**DT 17**

Stochastic profiles are in **DT 18** and **DT 19**. Each profile has a unique name, data type (type of profile) and profile data (**DT 19**): value and its weight.

Stochastic_Profiles (**name_profile(P)**; type_of_profile)	**DT 18**
Stochastic_Profiles_data(*name_profile(F)*; value; weight)	**DT 19**

Workers are associated to specific resources in **DT 20**. Routing and storing information are based on the information in **DT 21 - DT 24**. **DT 21** involves a set of operations that are possible to process in the specific resource and also information about priorities. This table creates a routing map definition. Routes of entities from buffer (space) to specific Resource are in **DT 22**. The given operation and information about routing from resource to target buffer is in **DT 23**. These tables also contain storing information: identification of buffers from (or to) where entities are stored.

HR_Resource (*ID_resource (F)*; *ID_HR (F)*)	**DT 20**
Operation_Resource (*ID_operation(F)*; *ID_resource (F)*; priority)	**DT 21**
Table_Storage_Machine (*ID_resource (F)*; *ID_operation (F)*; *ID_space (F)*; *ID_entity (F)*; *ID_crates (F)*; variant; order)	**DT 22**
Table_Machine_Storage (*ID_resource (F)*; *ID_operation (F)*; *ID_space (F)*; *ID_entity (F)*; *ID_crates (F)*; variant; order)	**DT 23**

Results from simulation model are recorded in tables **DT 24** - **DT 28**. Table **DT 24** contains the utilisation of each resource, such as busy, waiting, blocked, setup, repair times, etc. The waiting time is divided into several groups such as waiting for inputting entities, for labour, for free space in the output buffer, etc. Table **DT 25** contains results for each operation: the number of operations on each resource and the total time for each operation.

Results for buffers are in **DT 26** and **DT 27**. Information about each buffer and maximum values of each Entity, Crate, route number and variant is recorded. Simulation model involves some specific functions giving the following information about buffers: average time spent by entities, average buffer size (and occupancy), and also time spent while maximum occupancy was effective. These values describe buffer behaviour based on average or maximum values. For deeper analysis, it is possible to record complete historical data of all buffers containing arrival time and number of entities in the buffer.

Table **DT 28** incorporates throughput amounts of specific entities.

Resource_results (*ID_resource (F)*; busy time; blocking time; waiting part time; waiting space time; waiting off-shift time; setup time; repair time; waiting labour time)	**DT 24**
Resource_results_operation(*ID_resource(F)*; *ID_operation(F)*;	**DT 25**

number_of_operation; total_time)	
Space_results (*ID_space (F)*; *ID_entity (F)*; *ID_crates (F)*; variant; order; maximum_value; average time; average size; time_of_max)	**DT 26**
Space_results_behaviour_data (*ID_space (F)*; time; number_of_entities)	**DT 27**
Throughput (*ID_entity (F)*; *ID_crates (F)*; variant; order; number_of_entities)	**DT 28**

IDS database embeds some analysis as clustering of production into groups. Results are written in **DT 29** and **DT 30**. There is a list of unique clusters names in **DT 29** and in **DT 30** there are groups of resource for each cluster.

Clusters (**unique_Cluster_name(P)**)	**DT 29**
Clusters_data (*unique_Cluster_name (F)*; *ID_resource (F)*)	**DT 30**

Results and data from CAD layouts are in tables **DT 31** - **DT 35**. Table **DT 31** contains layers that are helpful tools to distinguish objects in the layout (filtering them). In IDS, each entity, crate or object type is involved in the specific layer in different colours.

Coordinates (X, Y, and Z) of important points are recorded in table **DT 32**. The important points are Input/output points of resources and space and they have unique names. **DT 33** contains all the information about facility layout and involved objects (resources, buffers, entities in buffers), CAD references used and inserted information (position coordinates, scale and rotation of inserted CAD blocks).

Material flows information are divided into ideal (straight) flows and real (constrained) flows in two tables (**DT 34** and **DT 35**). Flows are described with start and final coordinates (*unique point name)*, flow values, entity, crate, route order and variant.

CAD_layers (**unique layer name(P)**; colours_definition; layer_type)	**DT 31**
CAD_coordinates_points (**unique point name(P)**; coordinate X; coordinate Y; coordinate Z)	**DT 32**
Object_coordinates (object name; *CAD block reference(F);* coordinate X; coordinate Y; coordinate Z, rotation angle; scale x; scale y, scale z; *unique layer name(F)*)	**DT 33**

Material_flows_ideal (*unique point name*(F): from_point; *unique point name*(F): to point, *unique Entity name(F)*; *unique Crate name(F)*; variant; order; material_flow_value, *CAD layer(F)*)	**DT 34**
Material_flows_real (*unique point name*(F): from_point; *unique point name*(F): to point, *unique Entity name(F)*; *unique Crate name(F)*; variant; order; material_flow_value; *CAD layer(F)*); point_polyline_order; coordinate X; coordinate Y; coordinate Z)	**DT 35**

Factory design is based on the assignment of departments and selection of the optimum alternative. Departments are given by a unique name (**DT 36**) and they can easily be distinguished in the layout by colours. The definition of alternatives (**DT 37**) consists of a unique alternative, the factory space (width, length), the number of departments and the maximum width of departments. There are assignment algorithms that are used, such as CORELAP and CRAFT. CRAFT is based on material flows and costs between departments (table **DT 39**) while CORELAP is based on relationships with AEIOUX coefficients or alpha coefficient (table **DT 40**).

Both methods need a sequence of departments (**DT 41**) according to the department that is placed in the layout (**DT 42**).

Designed alternatives are saved in **DT 38** where, for each alternative, a department, a sequence of departments and a fixed department are saved. Table **DT 43** contains coordinates of departments for each alternative.

Departments (**unique_department_name(P)**; description; CAD colour)	**DT 36**
Alternatives (**unique_alternative_name(P)**; width; length; number departments; max_width_of_department; AEIOUX coefficients; alpha coefficient)	**DT 37**
Department_alternative (*unique_department_name (F)*; *unique_alternative_name(F)*; size, *Sequences(F)*; fixed)	**DT 38**
CRAFT (department_from; department_to; *unique_alternative_name*(F); flow_value; flow_cost)	**DT 39**
CORELAP (department_from; department_to; *unique_alternative_name*(F); relationship_value)	**DT 40**
Sequences (**unique_sequence_name(P)**)	**DT 41**
Sequences_data (*unique_sequence_name*(F); *unique_department_name* (F); order)	**DT 42**
Department coordinates (*unique_department_name* (F); *unique_alternative_name*(F); coordinates X; coordinates Y; coordinates Z)	**DT 43**

Database also involves transport and supplying systems - described in detail in Section 4.7. For each transport operation, it is defined "from station" and "to station" (final destination) (**DT 44**). Milk-run systems have specific behaviour so it requires a definition: a sequence of operations ("cycle") and also a cycle of each entity ("loading cycle") describing exchanges in the load.

Transportation_operation (*ID_resource* (F); from_station; to_station)	**DT 44**
Logistic_train (*ID_resource (F)*; cycle; loading_cycle*; ID_operation (F)*)	**DT 45**

Figure 49 displays Entity-Relationship Model (ERM). ERM is a conceptual graphical representation of database structure. Based on this model, database can be built in some application, as MS Access in the case of IDS.

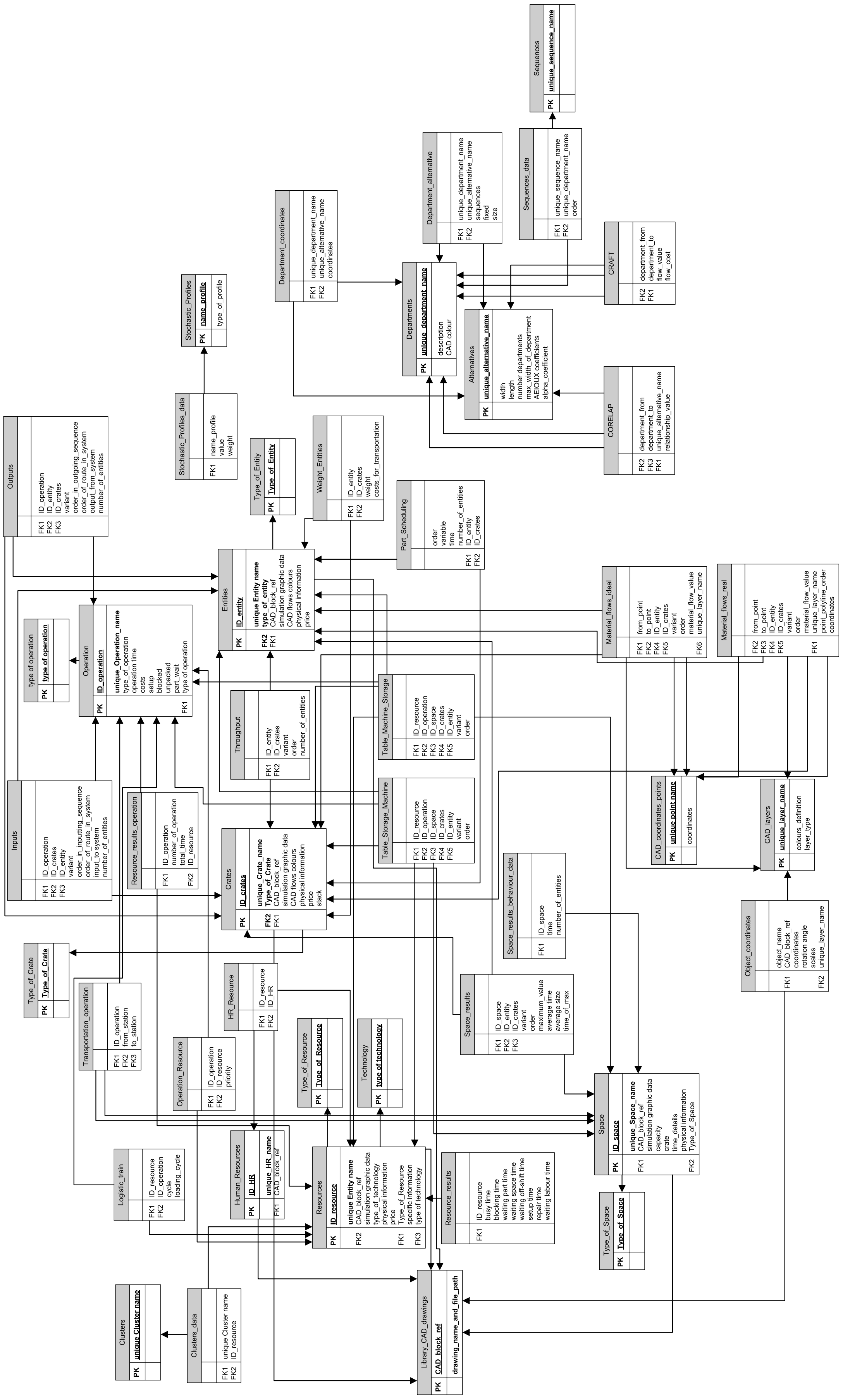


Figure 49 – Entity-Relationship Model

3.2.4 Simulation concept

Based on previous studies (see Section 2.3.2 - Discrete Event Simulation tools), a list of the most frequent tasks solved by computer simulation was established. A summary can be found in Table 8 – Frequent tasks for simulation. IDS would involve most of those mentioned tasks.

Background simulation

In the case of IDS, generated simulation models are primarily focused on running experiments and providing results to DB for further analysis. Therefore, simulation can be run in background of IDS system, completely independent on user´s interference. This approach has the following advantages:

- Users don't need to have a deep knowledge about simulation and they are able to create and use quite complex models just based on controlling data in the DB.
- Rapid design support

General model

As mentioned above, the simulation model should be helpful for any general type of problems. The generating procedure should apply to other simulation tools after code adaptation. Due to these reasons, there will be four **basic elements** involved in every simulation tool and, thus, making it possible to build any model: entity, resource, buffer and human resource. Crate is taken as an attribute of the entity (see Table 11) (SRobinson, Brooks, Kotiadis, & van der Zee, 2010).

Some specific objects, such as "transport carrier" that is supported by all of the simulation tools, are not used, so IDS will not make a relationship in these types of objects.

These objects will be created from the mentioned four basic elements. For example "conveyor" object can be created from a buffer with a defined minimum holding time.

These general objects and functions can be used in different simulation tools; the main principle remains (generation of objects, connection to DB and data load by SQL queries).

A general approach also provides a great number of variables that can allow configuration of the simulation model. All object properties and values are involved in DB and they can be changed for simulation experiments.

3.2.5 CAD concept

CAD system, that will be integrated into IDS, should provide the following most common tasks - these tasks are based on the analysis summarised in Table 8 and also on the analysis of similar CAD systems (Section 2.4.2):

- Factory design methods
- Layout design and spatial validation (position of equipment both for system and subsystem)
- Space for system and for subsystem (buffers)
- Automatically generated layout patterns
- Displaying material flows
- Optimization of layouts based on material flows and transportation costs
- Design of transportation system – transportation network design and finding the shortest path between stations

Factory design methods

IDS provides two basic placement methods for factory design: one based on the optimization of material flows (CRAFT method, see Section 2.2.1) and one method based on relationships (CORELAP). These two methods are chosen according to different approaches in the designing - general and universal use and also the possibility to use them simultaneously.

Layouts patterns

This very common task in the facility design can be automatically processed. IDS provides the automatic generation of three basic layout patterns: production line, cellular and process layout. Another helpful function for facility design is the spatial validation of buffers.

Classification of material flows

In IDS, several types of material flows are used. Table 12 shows the classification of flows. Material flows of a specific product display complete routes of it in the layout. These types of diagrams are called "Spaghetti Diagrams". It can be helpful as a basic identification of possible reduction and it can lead to layout improvements. Similarly, flows can be displayed for specific crates, e.g. pallets. Based on it, it is possible to easily design material handling and transportation system, e.g. forklifts aisles.

Flows can also be displayed as straight lines between starting and ending positions or as constrained polylines that were taken into account - aisles shapes, building or other limitations, etc.

Intensity flows are based on the number of parts through the given points per time unit. Intensity values can be obtained by multiplying intensity floes by product weight or flow costs:

- Intensity – number of parts per time unit [number of parts / day]
- Weight – number of units x unit weight per time unit [number of parts * kg / day]
- Cost – number of units x unit transportation cost per time unit [number of parts * € / day]

Flows between the same starting and ending positions can be summarised for all products or crates to see total intensities.

Transportation flows are summarised flows for every segment of transportation net. Transportation network consists of aisles, roads, etc. It can be helpful to analyse traffic congestion.

Table 12 - Flows Classification

Type of flow	Notice / Types
Based on part	Display complete flows of specific part
Based on crate	Material handling and transportation system design
Based on shape	Ideal (straight)
	Constrained (space limitation)
Type of intensity	Intensity flows
	Weight (total throughput x weight of each entities)
	Cost (based on unit transportation cost)
Summarised flows	Summarised flows between resources
Transportation flows	Summarised intensity flows for each segment of transportation network

3.2.6 Drawbacks and concept weaknesses

Proposed concept can involve drawbacks and weaknesses that are necessary to overcome. Main problems can be input data processing, interpretation of results, implementation of a real layout constraint and building a concept model considering simplifications (Mecklenburg, 2001) (Aleisa & Lin, 2005).

As mentioned before, IDS will support automatic processing of some facility layout phases that have also some disadvantages, such as the **loss** of user´s **continuity** during design.

As in similar solutions, the greatest problem is the implementation of **real layout constraints** as structural, ergonomic and other requirements in detailed layouts. It can be eliminated by the possibility of interferences based on user’s knowledge.

Other possible weakness is a running **background simulation** (see more in Section 3.2.4) and its typical drawbacks such as **graphic** (icons, animation, etc.) absence in the model. Animation can be very important for understanding the production system behaviour and also for presentation of the results and for communication with problem owner.

Automatically **generated models** have some differences to **manually made models**. Manually made models are directly focused on solving given problems. Automatically generated models can have a usage limitation - it is only possible to use them for specific tasks. For some other project tasks, it is not possible to use.

Human resource behaviour is the most difficult to describe. In IDS, the human resource implementation omits issues such as the scheduling of human work, multi-operation control, cooperation of many workers on one operation, skills and management and other very detailed properties. Workers logic is only based on allocation of the given operation to the resource and waiting for the workers. Other simplified issue is the use of shifts that have insignificant influences on the layout.

Data source issue can be also problematic. Each company uses a different database structure, different data sets, and a different type of database or even different data meanings. In IDS, there is an effort to design a general structure of database. This project database is loaded with data from a company database (e.g. ERP system). It is processed, creating a universal integration interface for the unification of both types of data structures (companies and IDS). For automatic loading, it can be very time-consuming and it is convenient in cases of complex and long projects.

IDS doesn´t support a complete integration of applications and changes based on feedbacks - e.g. manually added objects into a simulation model will not have an effect because those added objects must be processed in the DB first, so that a new simulation model can be generated.

3.3 Summary

This chapter introduced challenges in the area of factory and facility design and planning. Based on it, concepts of IDS were described. Functions of each type of tool were established as well as ways on how to integrate tools. In the next chapter, the implementation of IDS will be described.

4 IDS development

In this chapter, specific software tools for IDS are chosen based on their functionalities and integration possibilities. There is a detailed description of the development of IDS concepts, such as the principle of integration applications, the database design, the automatic generation of simulation models and the methods used in the CAD system: factory design procedures, layout patterns generation, material flows generation or transportation system design. This chapter involves examples of program codes for mentioned functions. The effort is to demonstrate basics of algorithms and tools integration. Full codes are involved in Appendix 3 of this work.

4.1 Software tools selection

In this section, specific tools have been selected to be included in IDS.

4.1.1 General requirements for software tools

The fundamental requirements for the chosen tools are to support integration in the meaning of **data exchange** and the direct **control** of one application by another. Applications must involve inner programming interface (e.g. macros) to create specific user functions. Convenient is to select tools with the same programming language. Tools should be widespread, stable and commonly used.

4.1.2 Selection of Database system

Database system (server) is a very important part of IDS. It was chosen from the following candidates:

- MS Access (www.office.microsoft.com, 2011)
- MS Excel (www.office.microsoft.com, 2011)
- Firebird (www.firebirdsql.org, 2011)
- PostgreSQL (www.postgresql.org, 2011)
- MySQL (www.mysql.com, 2011)
- Oracle (www.oracle.com, 2011)

The main disadvantage that rejects some of them is the high price (Oracle, around 5000€). MySQL, Firebird and PostgreSQL were rejected due to absence of application control devices, focus only on database functions, requirement of high level knowledge about database configuration, difficulties in programming user-defined functions, etc.

One of the serious candidates was MS Excel. The reasons are data visualisation and its processing (graphs, histograms or using a specific analysis functions already included) and very good solutions for importing data from external sources (data files, etc.). The drawbacks of MS Excel are the absence of SQL queries usage, slow searching of information and limited numbers of records.

MS Access (2010 version) was chosen for the following reasons:

- widespread of MS Access; almost on every computer with Windows operation system and Microsoft Office package
- price of license and service fees
- simple and effective work in design of SQL queries
- built-in Visual Basic for Application (VBA) to create functions (see Section 4.2.4)
- possibility of creating Input/Output forms

4.1.3 Selection of Discrete Simulation Tool

There are quite a lot of simulation tools with similar properties. Some simulation tools were mentioned in Section 2.3.2.

Candidates to IDS projects are:

- Arena (www.arenasimulation.com, 2011)
- Simio (www.simio.com, 2011)
- WITNESS (www.lanner.com, 2011)

WITNESS Simulation tool (PwE 2.01 version) was chosen for the following reasons:

- The possibility of control by another application to run simulation models in background
- The possibility to generate complete model automatically
- easy access to the DB by SQL queries
- author´s experience (almost 7 years)
- widespread in many companies (3rd in Simulation Tools ranking (Dias et al., 2011))
- stability
- possibility to involve another WITNESS tool in the IDS (WITNESS Scenario Manager, WITNESS Visio, Optimiser, WITNESS Virtual Reality etc.)

Another interesting solution can be to use some **free simulation** tools as "*Simulant*" currently developed at the University of Prague (www.simulant-addin.webnode.cz/, 2011) or similar add-in from Texas University (www.me.utexas.edu/~jensen/ORMM/, 2011), both of them are add-ins for MS Excel. These tools provide simulation functions that can be enough for the IDS as background simulation. Drawbacks are application stability, absence of some control logic functions and the fact that there are still under development.

The principle of automatic model generation and simulation approach can be used also in other mentioned tools – e.g. in Arena that have similar properties as WITNESS.

4.1.4 Selection of CAD system

CAD system is primarily used for layout drawings involving placement of resources, material flows, storages, transportation system, etc.

CAD systems considered as candidates are:

- AutoCAD (www.autodesk.com/autocad, 2011)
- Solid Edge (www.solidedge.co.uk, 2011)
- 3D Studio Max (www.usa.autodesk.com/3ds-max/, 2011)

AutoCAD (2011 version) was chosen for the following reasons:

- widespread tool, almost a standard for layout design
- possibility to make own functions inside
- wide range of functions and features

In AutoCAD application, two possibilities can be used to make own programs and functions – by AutoLISP or by AutoCAD ActiveX/VBA interface. The AutoCAD ActiveX/VBA interface provides several advantages over the other AutoCAD API environment:

- ActiveX applications (VBA) are faster than AutoLISP applications (see Section 4.2.4)
- Windows interoperability, ActiveX and VBA are designed to be used with other Windows applications and provide communication across applications, in the case of IDS - connectivity to the external DB
- The rapid development of VBA functions provides the perfect environment for prototyping applications, even if those applications will eventually be developed in another language
- Programmer base - millions of programmers around the world already use Visual Basic. AutoCAD ActiveX/VBA opens up AutoCAD customization (Sutphin, 2006)

There are also a great number of free CAD systems e.g. FreeCAD, (www.freecad.com, 2011) or SolidEdge (www.solidedge.co.uk, 2011) but with less functions, problems with stability, drawing format supporting, operation system, missing internal programming language, etc.

4.1.5 Additional software tools

There can be used also other tools that are not fully-integrated in IDS. These tools are helpful in simple works:

- **MS Excel,** for preparing some tables that are copied into DB (using some function as fast data copy, string function, import and export data from different sources, etc.). MS Excel is also used for cluster analysis in order to see each cluster as binary arrays. This application is very helpful in analysis and graph plotting (e.g. buffer)
- **Google Warehouse** – internet database of free 3D models available for AutoCAD usage (www.sketchup.google.com/3dwarehouse/, 2011)
- **Google Earth** (www.earth.google.com, 2011) – for using maps, measuring distances etc.

4.2 IDS overview

In the previous Chapter, concepts of integration database, simulation and CAD system is described. In previous sections, specific tools were described for each part of IDS. Figure 50 shows the overall integration and embedded functions. Integration approach is fully described in this section, as well as functions.

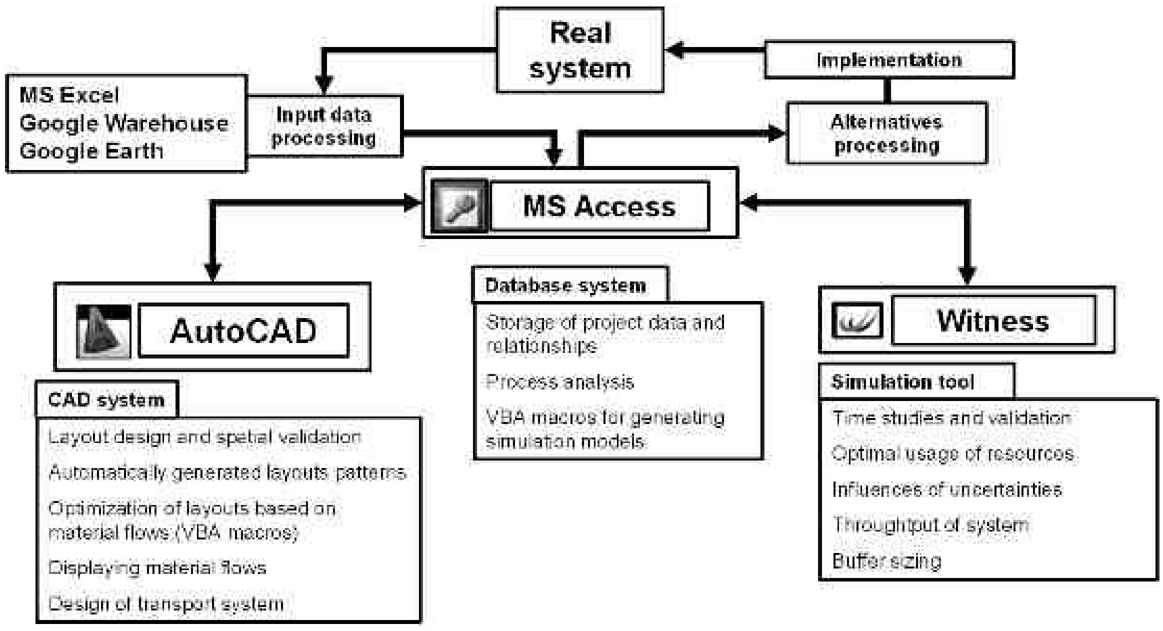

Real system
Implementation
MS Excel
Google Warehouse
Google Earth
Input data processing
Alternatives processing
MS Access
AutoCAD
Database system
Storage of project data and relationships
Process analysis
VBA macros for generating simulation models
Witness
CAD system
Layout design and spatial validation
Automatically generated layouts patterns
Optimization of layouts based on material flows (VBA macros)
Displaying material flows
Design of transport system
Simulation tool
Time studies and validation
Optimal usage of resources
Influences of uncertainties
Throughput of system
Buffer sizing

Figure 50 - Full integration solution and summary of functions

4.2.1 Applicable methods based on available data

The use of the IDS functions is highly dependent on the data. In Table 13, there is a list of the possible functions for each data set, labelled from (1) to (10).

A list of departments or resources is necessary for all methods used in IDS. This information can be joined also to department sizes. Departments are considered for factory design and resources for facility design. Manual placing of generated CAD blocks is possible in design layout (1) only if this list is available.

Table 13 – Applicable methods based on available data

(Sign) / Data set	List of department or resources	Material flows	Relationships	List of entities	Technology	Sequence of operation(tecnology)	Schedules	Operations	Set of operation	Space (Buffers) - storing information	Crates	Sequence of delivering cycles	Buffer occupancy	Constrained flows design CAD	Possible methods
(1)	x														manual placing
(2)	x	x													CRAFT, Graph Theory placement,
(3)	x		x												CORELAP
(4)	x			x											P-Q analysis
(5)	x			x	x										job-shop layouts
(6)	x			x		x									Cluster analysis, Production line design
(7)	x			x			x	x		x			x		Simulation
(8)	x			x			x	x	x	x			x		Simulation
(9)	x			x			x	x		x	x		x	x	Simulation, Transportation system design (forklifts)
(10)	x			x			x					x	x	x	Simulation, Transportation system design (milk-runs)

Material flows can be used as deterministic values, as in the case of (2), where *from-to chart* is used. Based on it, **factory placement methods** as CRAFT or Graph Theory Block Layouts are possible to process in these cases, without considering different entities. CORELAP (3) needs *relationship chart* between departments or resources. Results of (1), 2) and (3) are optimised layouts based on material flows or relationships and coordinates of placement of each of them.

Facility layout is possible to be designed based on P-Q analysis (4) for that data as list of entities and their production volumes. If the specific production equipment is known (5), job-shop layout or cellular and production lines can be generated (6). For that, it is required a list of entities and also technology sequence, routes of entities through resources (e.g. production machines). Cluster analysis helps to establish groups of resources and entities.

Operation data set is required for creating and running simulation models. Operation data are associated with storing information and with a set of operations for a given resource. It means "from which buffer" entities are pulled and "to which buffer" entities are pushed due to processing of given operation on a given resource (7). If there are more possible operations for the given resource or one operation can be processed on more resources, routing map of entities can be received (8) with all possible material flows.

Simulation experiments can give the following information: the usage of resources, buffer sizing, material flows based on stochastic influences, bottle-necks of systems, routing map, etc.

Transportation system is possible to design by simulation (time studies) and in the CAD (transportation aisles and roads). Based on different type of transportation, e.g. forklift for pallets (9), there are used different transport crates. A special case of transportation system is a milk-run system (10), where delivering cycles are modelled and analysed.

4.2.2 Table of objects

As mentioned above, in IDS, there are different applications that are used and so the same objects have different meanings and different types of representation. For example, a resource of objects is represented by a record in a **table** in the database, by a **machine** object in the simulation and by a **drawing** in the CAD system. In Table 14, there are basic system objects (resources, space, etc.), relationships or their properties and their expression (representation) in each software tool. Some objects are not involved in all applications, e.g. operation or production schedules are not represented in the CAD system; coordinates of objects are not important in the simulation model.

4.2.3 Integration technologies

As mentioned above, applications in IDS are integrated by the database, by **data exchanging** and by the **control of one application by another**.

For this purpose, the OLE (*Object Linking and Embedding*) technology is used. OLE is an integrative technology for sharing information by DDE (*Dynamic Data Exchange*) between programs, keeping the structures. If the object is transferred with the help of OLE, the link between both programs is kept, so it is possible to have an automatic updating of the embedded object after its change in the original program (Ševčík, 2002).

In IDS, there are two OLE approaches used - *OLE DB* and *OLE Automation*. **OLE DB** (Microsoft JET OLE DB Provider) is an interface for database access. Data exchanging is managed by loading/ saving data from the database by using SQL queries that, for instance can be sent from AutoCAD to MS Access as shown in the code in Figure 51.

Table 14 – Table of corresponding objects and their meaning

Objects, information, relationship	**DB (name of tables)**	**Simulation WITNESS Objects**	**CAD**
Entity	ListOfParts	*Part*	Drawing block
Resources	ListOfInventory	*Machines*	Drawing block
Space	ListOfSpace	*Buffer*	Rectangular area (or exact shape)
Human resources	ListOfHumanResources	*Labour*	Drawing Block
Crates	ListOfCrates	*Attribute for entity*	Drawing Block
Operation	ListOfOperation Inputs Outputs	Arrays	Not represented
Schedules of inputting parts	Tbl_Part_Scheduling	loaded from external Part files	Not represented
Stochastic profile	ListOfProfile	*Profile*	Not represented
Objects coordinates	CAD_basic_points_coordinates	Not represented (Tbl_Coordinates_Simulation_objects)	X,Y,Z coordinates, rotation, scale
Routing	Tbl_working_possible Table_Storage_Machine Table_ Machine_Storage	Arrays	Material flows
Material and information flows	Tbl_flows_ideal Tbl_flows_real	Arrays of flows in specific units (weight in time)	Polylines in different layers and in different colours
Transportation	tbl_operation_from_to tbl_from_to Tbl_Logistic_train	Arrays (transportation times, based on speed)	Paths by aisles (= polylines with nodes)
Layout patterns	ListOfClusters	Not represented	Position of each object
Factory design	Table_Alternatives Table_Department Table_Relationship	Not represented	Position of each object

```
Dim oAccess As New ADODB.Connection
Dim oRecordset As New ADODB.Recordset

'***project database location:

OutputStr=" D:\\disertation_work\\database_graphic_9.mdb"
oAccess.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & "Data Source=" &
OutputStr & ";"

'***SQL query and work with Recordset:

strQuery = "SELECT CAD_basic_points_coordinates.name_pnt, FROM
CAD_basic_points_coordinates;"
oRecordset.Open strQuery, oAccess, adOpenKeyset, adCmdText
With oRecordset
Do While Not .EOF

strName_point(i_i) = !name_pnt
…
```

Figure 51 – OLE DB example

The control of one application by another can be managed by **OLE Automation** which is a way of letting programs control each other. In IDS, OLE Automation is used for the automatic generation of simulation models from MS Access by Visual Basic functions. WITNESS is "*OLE*

Automation Server" and a Visual Basic function in MS Access is "*OLE Automation Controller*", which means that it can control other programs. An example of this kind of control is in the code in Figure 52.

```
'***Object WitObj substituting WITNESS in the Visual Basic code
Set WitObj=GetObject(,"WITNESS.WCL")
'***Order for simulation running and stop:
WitObj.Run(1000#)
WitObj.Stop

'***Set of object quantity:
WitObj.Action("SET QUANTITY ASSEMBLY 2")
'***Receiving statistics information busy time by WITNESS function "Putil"
of resource Drill_machine
strExp = "Putil(Drill_machine,2)"
busy_time = witobj. Expression (strExp)
```

Figure 52 – OLE Automation example

4.2.4 Programing language (VBA)

VBA (*VISUAL BASIC FOR APPLICATION*) supports programming of its own applications within other applications – so called macros (e.g. in MICROSOFT OFFICE tools). The general purpose is to develop new user-defined functions under the main applications.

There are several kinds of "Basic" programming languages:

- **BASIC** – sequence programming language (as FORTRAN, PASCAL, C etc.)
- **VISUAL BASIC** (current version 2011 using .NET) – objected programming language (as VISUAL C, VISUAL JAVA, etc.); for programming separate applications (*.exe, *.dll); includes debugger
- **VISUAL BASIC FOR APPLICATION** (version 7.0) - it is included in almost all MICROSOFT applications and other programs running under WINDOWS operation system
- **VISUAL BASIC SCRIPT** (VBS) (as JAVA SCRIPT) – programming of www pages in the side of clients

VBA is embedded in all IDS integrated tools and also in many another Microsoft applications; it is not required to buy a special licence to use it.

VBA is an object oriented programming language thus the full application code can be consisted of short procedures. VBA can be used for any kind of application because of its universality and also simple syntax. The environment of VBA involves visual and graphical features and user control interfaces that are designed from the included objects, by the "drag and drop" method.

The creating of an application usually has a programming part and a graphical design of interface forms. It is done by using VBA language orders, objects and their properties that are changed by events (e.g. click on button). The complete information about VBA programming can be found in many sources (McFedries, 2007)(Stephens, 2005)(Evjen, Hollis, McCarthy, & Sharkey, 2005).

4.2.5 Unit system selection

In IDS, the following units are used by default:

- length units (resource dimensions, areas) = millimetre [mm]
- time units (simulation) = second [s]
- transportation (speed) = meter per second [m/s]

One millimetre is the basic length unit used in engineering and it provides appropriate accuracy. In cases where layout alternatives are compared, it is possible to use dimensionless approaches and objects in the layouts can be placed in the relative distances. This can be enough to compare the efficiency of material flows.

4.3 Database

The database is created based on the concept specified in Section 3.2.3, in the chosen application MS Access. In the following sections, there are snapshots from database; examples of data are shown in Appendix 4 (Section A.4.12) on the illustrative examples and Input/Output forms.

4.3.1 Database structure

This section describes the structure of tables and their relationships in MS Access. In Figure 53, there are snapshots of tables representing entities, crates, resources, space and human resources. The explanation of these basic objects is in Section 3.2.3.

Figure 54 contains snapshots of operation definition (table *"ListOfOperation"*). Each operation has two sections: inputting (table *"Inputs"*) and outgoing (table *"Outputs"*). These tables define sequences of entities and their changes during the operation.

Figure 55 displays a system configuration through related tables that describe the fundamental characteristics of the behaviour of the system: operations (table *"ListOfOpeation"*) are related to resources (table *"ListOfInventory"*) through table *"Tbl_working_possible"* to defined sets of operations on the given resource. Routing and storing information are given by relations between tables *"ListOfInventory"*, *"ListOfOperation"* and "*ListOfSpace",* in tables *"Table_Storage_Machine"* and *"Table_Machine_Storage"*, as designed in the database concept. Table *"Table_Storage_Machine"* contains data about from which buffers entities are pulled to the resources whereas table *"Table_Machine_Storage"* contains data about buffers to where entities are pushed from resources.

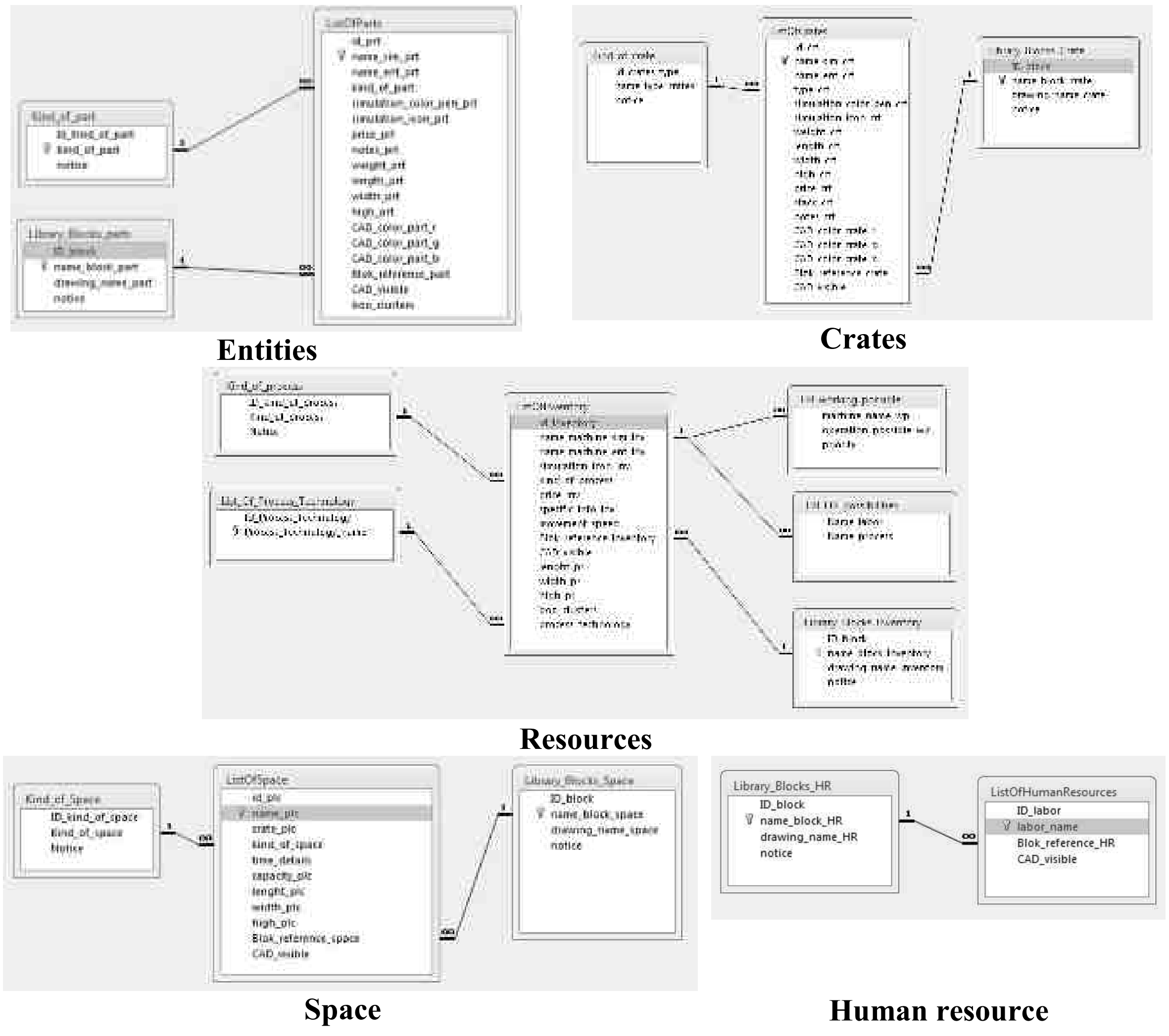


Figure 53 – Snapshot of relation database: basic objects

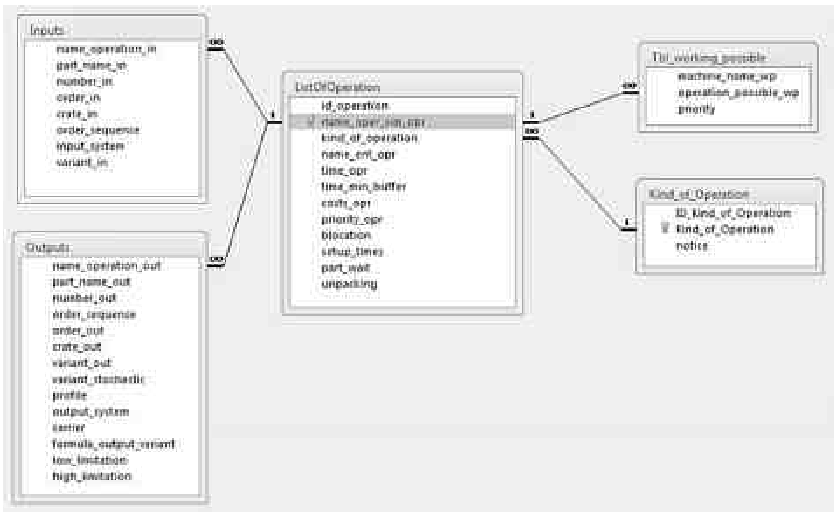


Figure 54 – Operation definition

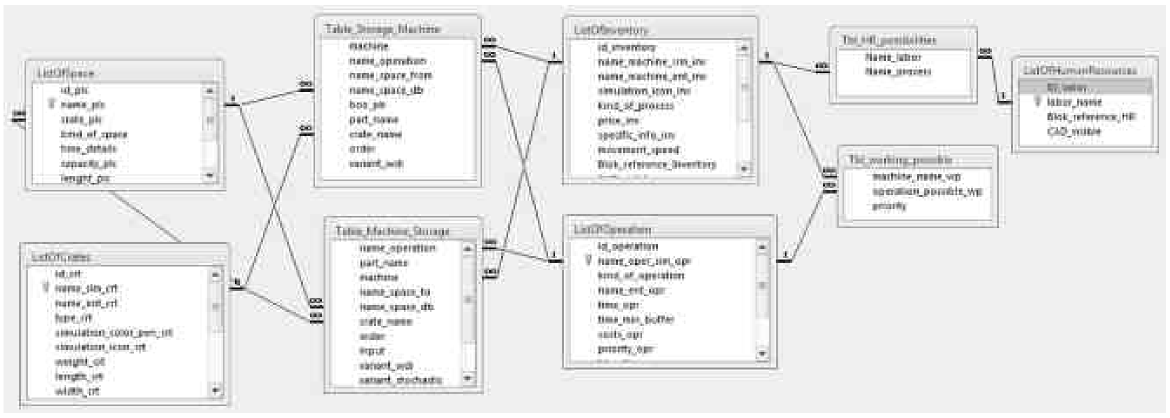


Figure 55 – System configuration

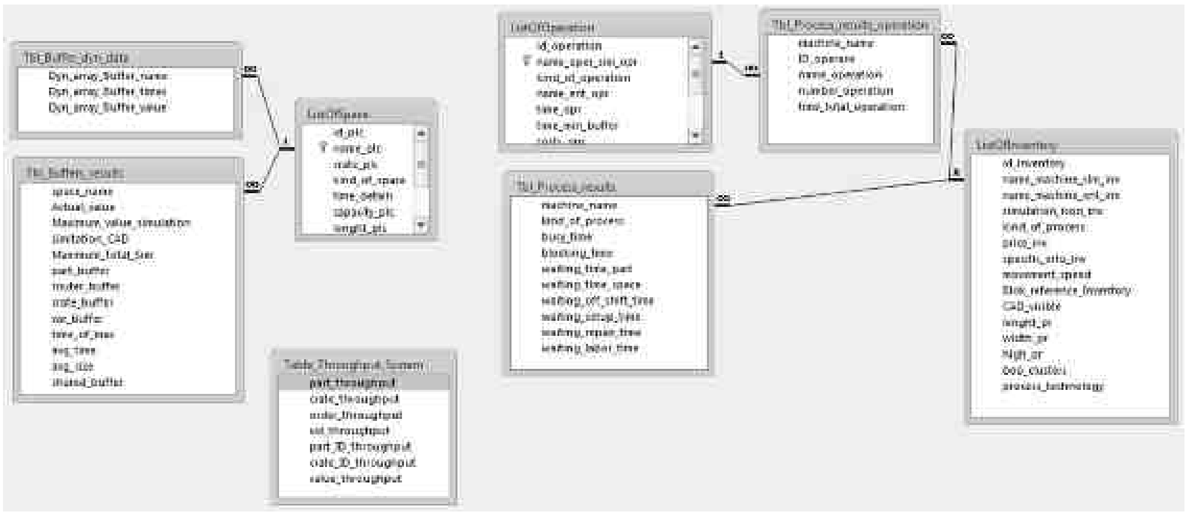


Figure 56 – Results from simulation experiments

Figure 56 displays the structure of tables concerning results from simulation experiments. Statistics of resources are involved in table *"Tbl_Process_results"* (statistics as busy, waiting times, etc.) and in table *"Tbl_Process_results_operation"* (results as number of operation for given resource, etc.). Buffers (space) statistics are saved in tables *"Tbl_Buffers_results"* and "*Tbl_Buffer_dyn_data"* as well as space limitation in different tools (CAD and simulation model).

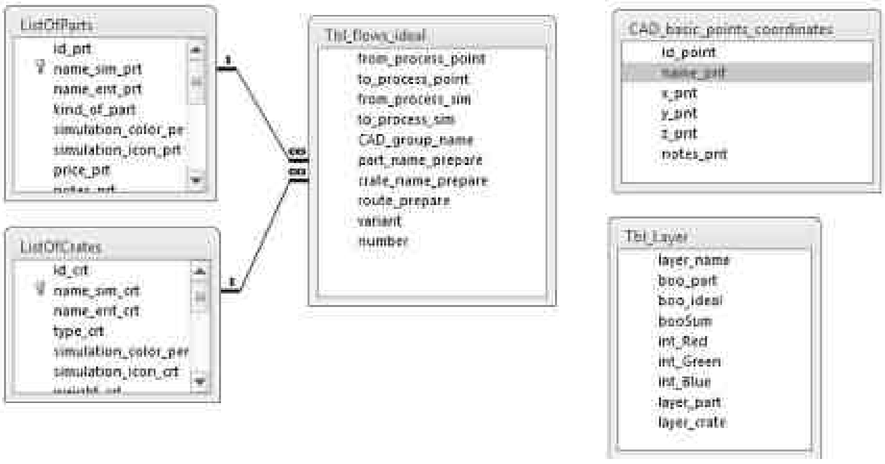


Figure 57 – Results from CAD layout

Results from the CAD layout are stored in the tables presented in Figure 57: coordinates of drawing blocks of the objects and data about material flows polylines.

Important tables and relationships for factory design are displayed in Figure 58.

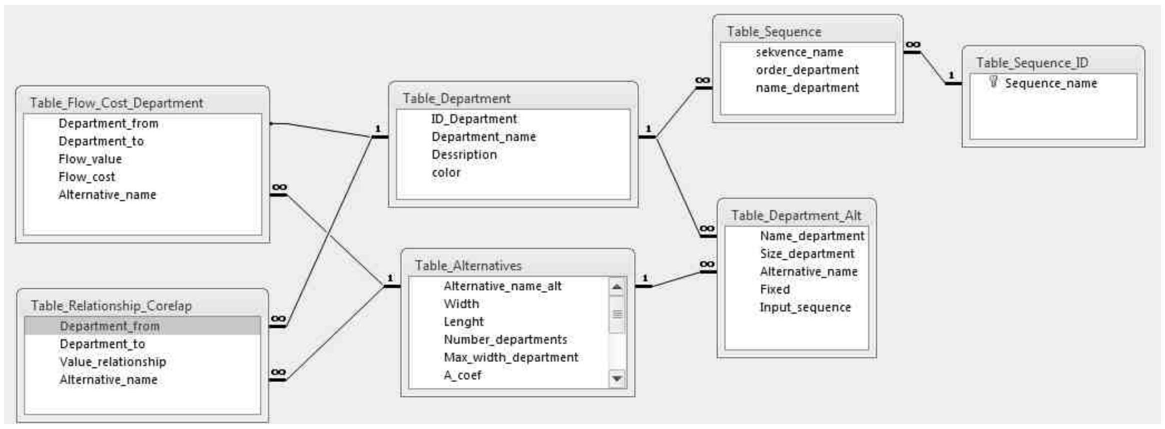


Figure 58 – Factory design

4.3.2 Implemented functions

Database in IDS involves several functions and analysis: the generation of simulation models, PQ analysis and Cluster analysis.

Simulation model generator

The generation of simulation models is described in details in Section 4.4 focused on the simulation.

P-Q analysis

It is a sorted array graphically represented by a plot graph or bar graph. The x-axis is a set of parts and the y-axis are numbers of incoming (planned) or outgoing parts (produced) from the system. Parts are sorted from high volumes to low volumes (see Section 2.1.1). The implementation makes use of a SQL query and a database object Microsoft Graph Chart (MSGraph.Chart.8). An example of SQL query is in Figure 59.

```
SELECT Table_Throughput_System.part_throughput,
Sum(Table_Throughput_System.value_throughput) AS SumOfvalue_throughput FROM
Table_Throughput_System GROUP BY Table_Throughput_System.part_throughput
ORDER BY Sum(Table_Throughput_System.value_throughput) DESC;
```

Figure 59 – SQL query example of PQ analysis

Cluster analysis

In IDS, there are two involved heuristic algorithms in cluster design: DCA (*Direct Clustering Algorithm*) and ROC (*Ranking Order Cluster Algorithm*). These two algorithms are well known and described in many works (see Section 2.1.2). The full implementation codes are included in Appendix (Section A.3.2). Examples of cluster design are in Section A.4.9.

4.4 Simulation

As mentioned above, the WITNESS Simulation Tool was chosen to be part of IDS. According to conceptual model, IDS uses an approach of Automatically Generated Simulation Models (**AGSM**)(see Section 3.2.4).

In this section, it is described the principles of AGSM, the specification of each object and the required functions for automatic generator and simulation running. In Table 14, there is a table of the corresponding objects and their meaning in the different tools.

4.4.1 Principles of simulation models generation

Simulation models are automatically generated from MS Access by a set of programmed functions (VBA macros). The process of generation and use of AGSM have the following phases:

- Data preparation
- Generation of objects
- Initialisation of model
- Experimentation
- Finalization

Figure 60 displays the complete process of generation, with points from (1) to (8) showing steps in logical order.

The generation itself starts with **loading** appropriate **data** from DB (1). It is loaded the following data: "List of Resources", "List of Entities", "List of Space", "List of Human Resources", "List of Operation", "List of Stochastic profiles", "List of Part arrivals" and information about output borders specification. Based on these data, it is possible to generate a complete simulation model.

Basically, the **Generation** of simulation objects is processed by sending orders from MS Access to a WITNESS Simulation model where they are executed and proper objects are created. The connection among applications is managed by WITNESS WCL library providing *OLE Automation Server*. It provides controlling WITNESS by another application.

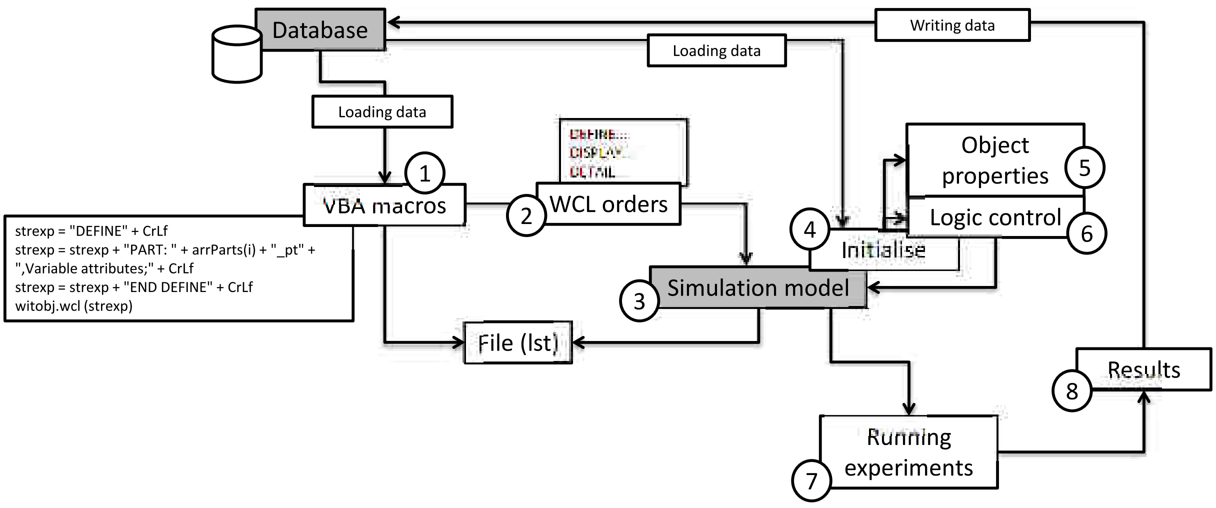
In MS Access, there is a set of functions for a full automatic construction of these orders. Orders are based on the WITNESS inner programming language called *"WITNESS Command Language"* (**WCL**). WCL orders (2) have a specific structure consisting of three sections:

- DEFINE
- DISPLAY
- DETAIL

Firstly, objects must be declared (DEFINE), then the graphic properties are generated into the model space: icons, colours, position in the model space, etc. (DISPLAY). Finally, logic properties of objects are defined: input and output rules, action on input and output, assignment of variables, etc. (DETAIL)(3).

Orders, or sequences of orders, are built from specific WCL functions – see example of code for creating basic elements ("Machine001") in Figure 61 (left). The full documentation for this function is in the WITNESS WCL manual. This WCL code is constructed by VBA macros, as shown it the same figure on the right. This macro prepares WCL code involved in string variable *"strexp"* that is sent and executed in the simulation model by the function *"witobj.WCL (strexp)"*, where "*witobj"* is an object supporting OLE application controlling. In the code example, it is used an array "*arr_machines(i)"* containing data from "List of Resources". Suffix "_pr" is added in the end to determine different types of objects in the simulation model.

By similar functions, all objects are generated – resources, buffers, human resources, entities and also all variables and arrays. Both variables and arrays will be described later.



```
strexp = "DEFINE" + CrLf
strexp = strexp + "PART: " + arrParts(i) + "_pt" +
",Variable attributes;" + CrLf
strexp = strexp + "END DEFINE" + CrLf
witobj.wcl (strexp)
```

Figure 60 – Principle of Automatic Model Generation

By VBA macros only the objects and their graphic are generated. The setup of object details is processed by **the initialisation** phase (4), which is run directly from the simulation model. In this phase, the required data are loaded from DB into the generated model and assigned into arrays and variables. Based on this data, the phase corresponding to DETAIL is processed. This phase is described in details in Section 4.4.6 (5) (6). It is more convenient to load data into simulation model once, instead of using SQL query to DB every time it is needed.

Then, AGSM is fully prepared for **running experiments** (7) and results can be written into the DB in the **finalization** of the model (8).

If the model is generated once, then it is not necessary to process all the initialisation phases because that model has been already configured. This is possible because only values of variables are refreshed, which enables skipping other phases as objects details specification.

```
DEFINE
    MACHINE: Machine001,1,General,0,0;
END DEFINE

DISPLAY
    SELECT
        Machine001
        NAME:#0,Name,Group,11,RGB(128,0,0),RGB(0,0,0),2,296,120,400,0,Aria,
        0,0,0,0,0,0,3,2,1,34;
        PART: #0,Part Queue,
        Member,Right,RGB(139,0,0),RGB(0,0,0),8,0,1,All,342,149,2;
        LABOR: #0,Labor Queue,Member,Down,RGB(139,0,0),RGB(0,0,0),0,-
        8,2,All,337,137,2;
        MACHINE ICON: #0,Main
        Icon,Member,RGB(255,255,255),1,296,135,5,4.85714285714,0,0,0;
        MACHINE ICON: #0,Status Icon,Member,Status,91,336,160,1,1,0,0,0;
        END Machine001
    END SELECT
END DISPLAY

DETAIL
    SELECT
        Machine001
        NAME OF MACHINE: Machine001;
        QUANTITY: 1;
        TYPE: General;
        * Input qty: pole (2,3);
        * Output qty: 1;
        * Inherit Attribute Values: No;
        PRIORITY: Lowest;
        LABOR:
        Cycle: None;
        END
        DISCRETE LINKS :
        Fill: None
        END
        CYCLE TIME: Undefined;
        INPUT RULE: PULL from Buffers001;
        OUTPUT RULE: PUSH to Buffers002;
        Output_From: Front;
        REPORTING: Individual;
        SHIFT: Undefined,0,0;
        END Machine001
    END SELECT
END DETAIL
```

```
…
Set witobj = GetObject(, "WITNESS.WCL")

For i = 0 To UBound(arr_machines)

   strexp = "DEFINE" + CrLf
   strexp = strexp + "MACHINE: " + arr_machines(i) + "_pr" + ",1,Single,0,0;" + CrLf
   strexp = strexp + "END DEFINE" + CrLf

   On Error GoTo Error_in_gen
   witobj.WCL (strexp)
   txtGen_sim = txtGen_sim + strexp
Next

For i = 0 To UBound(arr_machines)

                strexp = strexp + CrLf
                strexp = strexp + "DISPLAY" + CrLf
                strexp = strexp + "SELECT" + CrLf

                strexp = strexp + arr_machines(i) + "_pr" + CrLf
                strexp = strexp + "NAME: #0,Name,Group,-
11,RGB(128,0,0),RGB(0,0,0),2," + str(intStart_positionX_temp - 8) + "," +
str(intStart_positionY_temp - 16) + ",400,0,Arial,0,0,0,0,0,0,3,2,1,34;" + CrLf
                strexp = strexp + "PART: #0,Part
Queue,Member,Right,RGB(139,0,0),RGB(0,0,0),-8,0,1,All," + str(intStart_positionX_temp +
40) + "," + str(intStart_positionY_temp + 32) + ",2;" + CrLf
                strexp = strexp + "LABOR: #0,Labor
Queue,Member,Down,RGB(139,0,0),RGB(0,0,0),0,-8,2,All," + str(intStart_positionX_temp +
32) + "," + str(intStart_positionY_temp) + ",2;" + CrLf
                strexp = strexp + "MACHINE ICON: #0,Main
Icon,Member,RGB(255,255,255),1," + str(intStart_positionX_temp) + "," +
str(intStart_positionY_temp) + ",5,4.85714285714,0,0,0;" + CrLf
                strexp = strexp + "MACHINE ICON: #0,Status Icon,Member,Status,91," +
str(intStart_positionX_temp + 40) + "," + str(intStart_positionY_temp + 16) + ",1,1,0,0,0;" +
CrLf

                 strexp = strexp + "END " + arr_machines(i) + "_pr" + CrLf
                 strexp = strexp + "End SELECT" + CrLf
                 strexp = strexp + "END DETAIL" + CrLf

                 intStart_positionX_temp = intStart_positionX_temp
                 intStart_positionY_temp = intStart_positionY_temp + 100

Next 'i

…
```

Figure 61 – Example of WCL

Functions, attributes, graphic properties or model setup properties that are used in every model are involved in an external template file *("graphic.mod")*. Simulation objects are generated into these files. IDS supports two ways of simulation model preparing. The first way is direct generation of simulation model by WCL orders through OLE as described above. The second way is to generate complete WCL data for simulation model into external file ("*lst*") that can be imported to the WITNESS. Simulation model involves **functions** for initialisation model, loading data, configuration model, definition of objects, internal and external logic control, etc. Some of these functions are described in the following sections; the rest is summarised in Appendix (Section A.3.3.3) with the complete programme code.

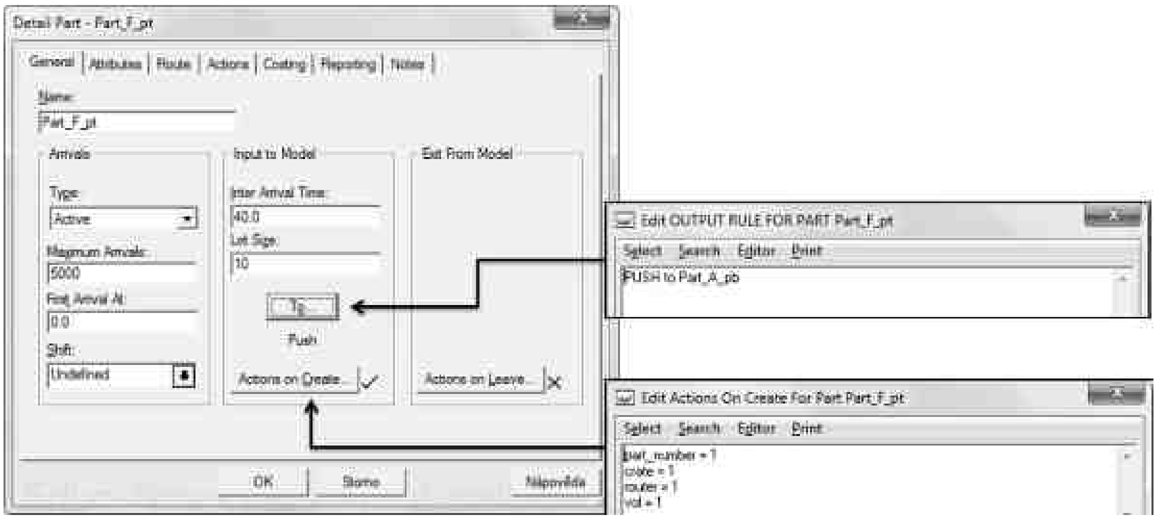
4.4.2 Entity object properties

Every entity is clearly defined in the simulation system by its name and attributes that are different for each step in its life. Table 15 summarised these attributes and gives examples. Attributes are integer or string values held by each entity itself. Attribute "PartNumber" can be changed on "Action on Output" (resource) in the operation where new entities are created, e.g. in assembly operation. "Crate" attribute is used e.g. in the unpacking kind of operation (simulation) and differentiation of material flows for transportation system. Value of "Router" is increased on "Action on Output" (resource).

Table 15 – Properties of entities

Attribute name	**Meaning**
"PartNumber"	unique number defined entity ("part_A", "screw" etc.). It corresponds to the "ID_entity" in the DB.
"Crate"	unique number defined crate ("pallet", "box" etc.)
"Router"	order number of the operation sequence. It is automatically increased on "Action on Output"
"Vol"	definition of product variants, quality level, free for use
"operation_list(router)"	array to which ID of processed operation are recorded
"routing_list(router)"	string array which visited resources names are recorded

Router numbers on resource input are even and on output they are odd. This kind of numbering is applied due to easy description of every operation type (assembly, cutting, multi-operation machine etc. or repairing operation where the same operation is processed again on the same resource).



Name: Part_F_pt

Type: Active
Maximum Arrivals: 5000
First Arrival At: 0.0
Shift: Undefined

Inter Arrival Time: 40.0
Lot Size: 10

```
PUSH to Part_A_pb
```

```
part_number = 1
crate = 1
router = 1
vol = 1
```

Figure 62 – Details of part object

The creation of entities

Entities can be created by three ways - directly from the simulation objects *Part*, from *Part file* definition or from resource as a final product of operation. Objects Part (see Figure 62) is generated for each entity and it can also be a source of entities. Entities are created based on the following parameters - maximum arrivals, first arrival time, interval between arriving and lot size. After the creation, attributes are assigned and entity is pushed to the next destination (*Output rule*). A part file is an external file involving arriving schedules. An example of this use is in Section A.4.2. The last possibility for entity creation is the output from the operation, e.g. during cutting operation different entities can be produced.

4.4.3 Resource object properties

Resource is a simulation object that represents production machines, transporters or fictive machines. Production machines can be CNC machines, assembly workstation, etc. Transporters are e.g. forklifts, trains, taxi, milk-run, etc. Fictive machines don´t physically exist, their function can be controlling material flows. Fictive machines can also be places where operations without requirement to any physical equipment are being processed, such as checking, unpacking, etc.

The general function of resources in the simulation model is to process *operations*. Operations change properties of entities that go through resources. Properties of entities can be part type, crate, order of operation, etc., as described in Section 4.4.2.

Each resource can be defined by the following set of logic control sections, variables and activities:

- Input/Output logic
- Input/ Output quantity
- Actions on Input/ Start/ Finish/Output
- Cycle time
- Labour Rule
- Setup and breakdowns

Logic control

Properties of resource and operation control are managed by a set of variables and functions, called *internal logic control*. An operation is selected for a given resource by *external logic control* that is based on priorities of a set of resources and operations, available quantity of inputting entities, available buffer space for outputting parts, etc. External logic principle is described in details in Section 4.4.7. Results of this external logic are assignments value "ID_operation" and names of buffers from which inputting entities are pulled into the resource. Arriving of entities into resource triggers *internal logic control*, which is described below. By value "ID_operation", the operation is uniquely determined.

Internal logic control

In Figure 63, there is a complete schema of the machine properties and variables that are automatically generated within the Initialisation phase. **Input rule** and **Action on Input** control inputting entities, pulling appropriate **input quantities** from buffers. After pulling of all entities to the resource, operation with specific ID number can start being processed. There are two sets of actions - **Action on the start** and **Action on the finish**. Between these events, the **operation time** is run. Based on **Output rules**, outputting entities are sent to specific buffers and then they trigger **Actions on Output**. The resource can have **breakdowns** and it can need time to be **setup**. For operation processing, specific **Labour** can be associated.

All those activities are described by variables, depending on the operation process or stochastic functions.

The complete principle of inner logic control of operation within resource is described in the following sections on the example of resource "Brush_1". Names of these variables are based on [Resource name] + [suffix], e.g. operation time of resource "Brush_1" is "Brush_1_tm". The list of all variables used for internal logic control is in Table 16. Complete codes and detailed explanation are in Appendix (Section A.3.3.3).

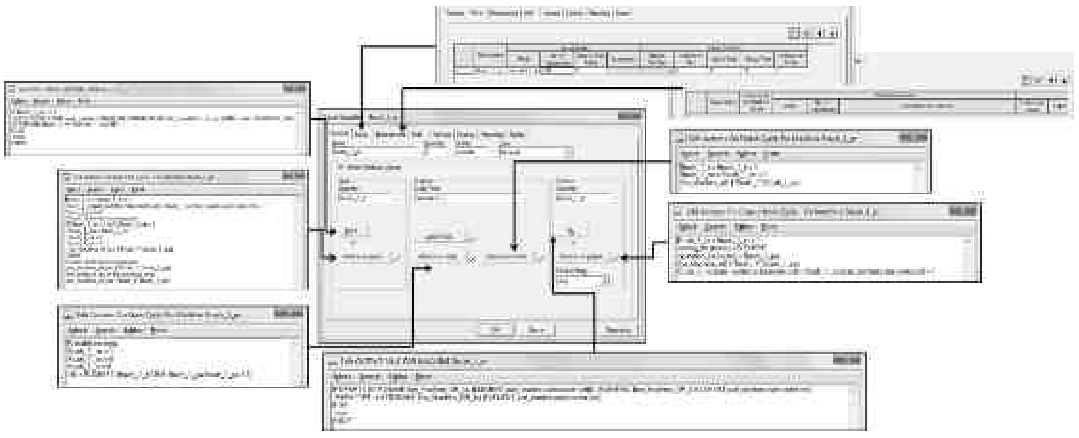


Figure 63 - Details of Machine objects from WITNESS

In this section, there is an explanation of the principle of internal logic control; the basic meaning of functions is also described. Inputting information of internal logic control is a value of "Brush_1_pc (ID operation)", "Brush_1_ow (operation name)" and "Brush_1_wh" containing names of buffers. Each operation is described by a set of arrays providing information about incoming/outgoing entities, their quantities and properties. Operations with more different incoming/outgoing entities (e.g. assembly) have arrays with second dimensions greater than one and variable "Brush_1_hp" holds the current row index of this array.

Table 16 – Properties of resource

Variable	Meaning	Declaration type
[Resource name][_**pc**] [Resource name][_**pa**]	ID operation	Integer
[Resource name][_**ow**]	operation name	String
[Resource name][_**wh**]	buffers	String
[Resource name] [_**hp**]	index of current operation row	Integer
[Resource name] [_**ic**] [Resource name] [_**oc**] [Resource name] [_**in**] [Resource name] [_**ot**] [Resource name] [_**ii**]	counters of inputting, outgoing entities	Integer
[Resource name] [_**tm**]	operation time	String
[Resource name] [_**ni**]	material flows in(part_number,crate,router,vol)	Integer
[Resource name][_**no**]	material flows out(part_number,crate,router,vol)	Integer
[Resource name] [_**tt**]	Full information about processed operation on given resource (temp time, ID operation, total time)	Real

a) Input logic

This logic controls inputting entities (see Figure 64). Whenever a value "Brush_1_pc" is changed by external logic functions and value "ID_operation" is assigned, the resource tries to find specific entities in buffers ("Brush_1_wh") by **matching attributes of entities**. Entities are determined by a set of attributes (*part_number*, *crate*, *router* and *vol*) thus only a correct entity can be pulled into the resource. Function *SNAMEVAL (Brush_1_ow)* gives a string value held in the variable "Brush_1_ow" (operation name), and *INAMEVAL (array, row, column)* gives an integer value from array.

```
IF Brush_1_pc <> 0
MATCH/CONDITION (part_number = INAMEVAL (SNAMEVAL
(Brush_1_ow),Brush_1_hp,1) AND router = INAMEVAL (SNAMEVAL
(Brush_1_ow),Brush_1_hp,3) - 1 AND crate = INAMEVAL (SNAMEVAL
(Brush_1_ow),Brush_1_hp,2) AND vol = INAMEVAL (SNAMEVAL
(Brush_1_ow),Brush_1_hp,5))
STR2NAME (Brush_1_wh (1,Brush_1_hp)) #(1)
ELSE
   Wait
ENDIF
```

Figure 64 - Resource: Input logic code

b) Action on Input

The Action on Input is triggered every time an entity arrives into the resource (see Figure 65). This information is saved in an array of material flows ("Brush_1_ni (part_number, crate, router, vol") whose specific value is increased. If the arriving part is the first, function "*fce_Machine_AI_first*" resets the counters of entities. Function "*fce_Machine_AI_last*" checks if incoming entity is the last from the sequence. If so, the operation time is assigned and the operation can start. Function "*fce_Machine_AI_inc*" checks the quantity of arriving parts of the current operation row. If it is achieved an accurate number of entities, the variable "Brush_1_hp" is increased.

```
    Brush_1_ic = Brush_1_ic + 1
    Brush_1_ni (part_number,crate,router,vol) = Brush_1_ni
    (part_number,crate,router,vol) + 1
    Brush_1_tm = 0
!Check of the first incoming part
    IF Brush_1_ic = 1 AND Brush_1_hp = 1
      Brush_1_pa = Brush_1_pc
      fce_Machine_AI_first ("Brush_1",Brush_1_pa)
    ENDIF
!Check of the last incoming part
    fce_Machine_AI_last ("Brush_1",Brush_1_pa)
!increasing of row in the operation array
    fce_Machine_AI_inc ("Brush_1",Brush_1_pa)
```

Figure 65 - Resource: Action on Input code

c) <u>Action on Start</u>

After all entities have arrived into the resource, some variables are reset and current simulation time is saved in variable "Brush_1_tt", in the row for given operation ID (see Figure 66). This information is helpful in statistics.

```
!Variable resetting
Brush_1_hp = 1
Brush_1_oc = 0
Brush_1_ic = 0
OK = RVARSET (Brush_1_tt,TIME,Brush_1_pa,Brush_1_pa,1,1)
```

Figure 66 - Resource: Action on Start code

d) <u>Action on Finish</u>

After the expiration of the operation time, Action on Finish is triggered (see Figure 67).

```
Brush_1_ii = Brush_1_ii + 1
Brush_1_oc = Brush_1_oc + 1
fce_Machine_AF ("Brush_1",Brush_1_pa)
```

Figure 67 - Resource: Action on Finish code

Function "*fce_Machine_AF*" contains a code that changes attributes of entities (see Figure 68). Attributes part_number, crate and router are given by the operation array. Attribute "*vol*" can be described by stochastic function, so its value can be variable, based on the value of profile (function *fce_Profiles ()*). For example, this attribute can be used for distinguishing the quality level of final products. Based on it, different output buffers can be defined.

```
…
part_number = INAMEVAL(name_operation,inameval(help_process_hp),1)
crate =INAMEVAL(name_operation,inameval(help_process_hp),2)
router = INAMEVAL(name_operation,inameval(help_process_hp),3)

!vol:
if INAMEVAL(name_operation,inameval(help_process_hp),8)=-1 !with
stochastik
    vol = fce_Profiles ()
else
    vol=INAMEVAL(name_operation,inameval(help_process_hp),7) !without
stochastic
endif
…
```

Figure 68 - Resource: "Function_Machine_AF" code

```
IF NPARTS (STR2NAME (fce_Machine_OR_bu
(ELEMENT,part_number,crate,router,vol))) < INAMEVAL (fce_Machine_OR_li
(ELEMENT,part_number,crate,router,vol))
  PUSH TYPE to STR2NAME (fce_Machine_OR_bu
(ELEMENT,part_number,crate,router,vol))
ELSE
  Wait
ENDIF
```

Figure 69 - Resource: Output logic code

e) Output logic

In Output logic section (see Figure 69), entities are pushed into buffers, in case the number of entities (function "*fce_Machine_OR_bu()*") is lower than the buffer limitation (function *"fce_Machine_OR_li()"*).

f) Action on Output

Within entities' leaving of resource, entity routing information is recorded: *"routing_list (router) = ELEMENT"* and "*operation_list (router) = Brush_1_pa"* (see Figure 70). This is helpful in analysis of material flows as well as the total throughput of given resource saved in array *"*Brush_1_no*"*. Function *"fce_Machine_AO()"* records the current time into *"*Brush_1_tt*"* and reset all internal variables.

```
Brush_1_ic = Brush_1_ic + 1
routing_list (router) = ELEMENT
operation_list (router) = Brush_1_pa
fce_Machine_AO ("Brush_1",Brush_1_pa)
Brush_1_no (part_number,crate,router,vol) =
Brush_1_no(part_number,crate,router,vol) + 1
```

Figure 70 - Resource: Action on Output code

4.4.4 Buffer object properties

Buffer object represents space and its basic function is holding entities until they are pulled into the next operation. Small examples shows an importance of buffers usage in production system (see Appendix 2).

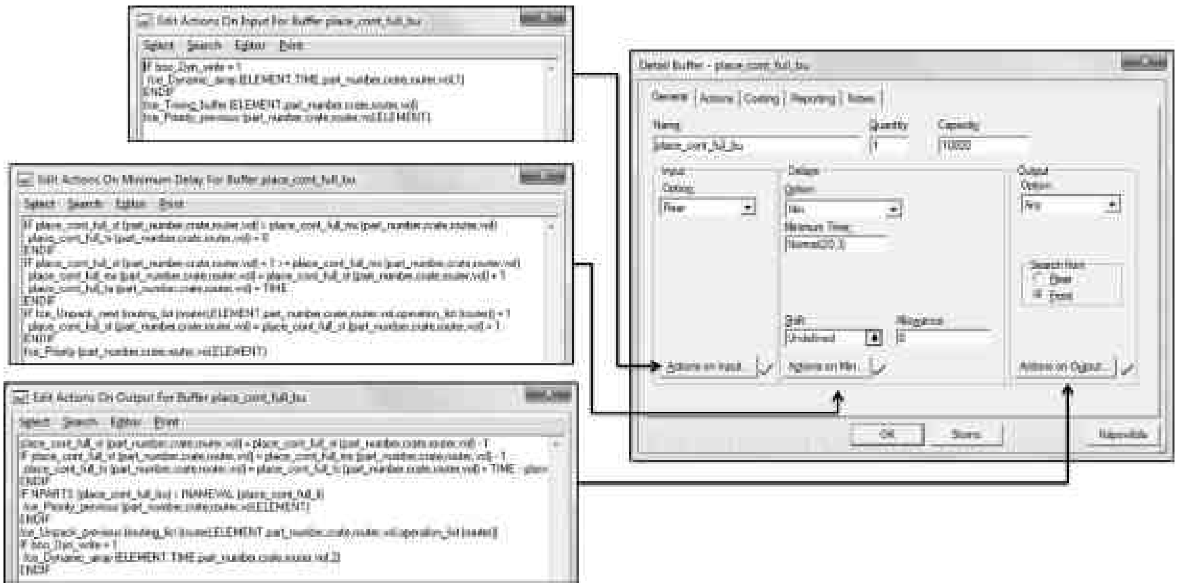

Edit Actions On Input For Buffer place_cont_full_bu
Edit Actions On Minimum Delay For Buffer place_cont_full_bu
Edit Actions On Output For Buffer place_cont_full_bu
Detail Buffer - place_cont_full_bu
General
Actions
Costing
Reporting
Notes
Name
Quantity
Capacity
Input
Option
Rear
Delays
Minimum Time
Output
Any
Search from
Front
Shift
Undefined
Actions on Input...
Actions on Min...
Actions on Output...
OK
Cancel

Figure 71 - Details of Buffer object from WITNESS

IDS uses buffers with defined delay time so that entities must stay there. This property is used in transportation where buffers represent conveyors or time paths (see details in Section A.4.2. It is also used in cases of production storages where resource and buffer create a unit that substitutes more resources of the same kind and properties. In IDS, there are two types of buffers – one with the mentioned properties and another one *"Input System Buffer"*, called "*prebuffer*". The list of variables describing buffer is mentioned in Table 17.

Table 17 – Properties of buffer

Variable	**Meaning**	**Declaration type**
[Buffer name][_**st**]	current number of entities inside buffer	Integer
[Buffer name][_**mx**]	maximum number of entities ever appeared in the buffer	Integer
[Buffer name][_**ts**] [Buffer name][_**ta**]	time of maximum occupancy	Real
[Buffer name][_**li**]	limitation of buffer	Integer

In Figure 71, there is an example of Buffer Details from WITNESS Simulation Tool.

As resources, buffers have also internal logic control. It consists of three sections; incoming entities trigger **Actions on Input**. If entities have a defined **Minimum delay time**, they stay there for at least this time. Entities wait in the buffers until they are pulled into resources. As soon as they leave the buffers, functions in **Action on Output** are triggered. The principle is described on the concrete buffer – “place_cont_full”.

a) Action on Input

Inputting entities trigger the following set of functions (see Figure 72). If there is a need for analysis of the complete behaviour of buffer within time, function *“fce_Dynamic_array()“* records all the information about entity and time into the specific array. Function *“fce_Timing_buffer()“* sets a minimum time delay up, if it is demanded. Function *“fce_Priority_previous()“* searches an operation for the resource from where the entity came. The principle of this searching is the basic of **external logic control** and it is described in detail in Section 4.4.8.

```
IF boo_Dyn_write = 1
      fce_Dynamic_array (ELEMENT,TIME,part_number,crate,router,vol,1)
ENDIF
fce_Timing_buffer (ELEMENT,part_number,crate,router,vol)
fce_Priority_previous (part_number,crate,router,vol,ELEMENT)
```

Figure 72 - Buffer: Action on Input

b) Action on Minimum delay

After the minimum delay time expired, the following set of functions is triggered (see Figure 73). The current number of entities is compared with the historical maximum. If it is necessary, the maximal value is refreshed in value of variable “place_cont_full_mx“, as well as the current simulation time. These values are used as simplified statistics: they give information about the maximum number of specific entities that appeared in the buffer and also the time period of this maximum. Function *“fce_Unpack_next”* checks if entities are products of an unpacking operation. Two entities are created by the unpacking operation – carrier and loaded items. Carrier (e.g. pallet) can be used in the next operation only after all loaded items (e.g. boxes) are taken away and after the carrier became empty. If the entity is not a carrier, the function returns to value 1 as well as the carrier is ready for the next operation. The number of entities in the buffer is then increased (see example in Section A.4.2). Function *“fce_Priority”* searches the next operation in order for the current entity.

c) Actions On Output

Functions involved in the “Actions on Output” are triggered whenever an entity leaves a buffer (see Figure 74). The current number of entities is decreased in the array “place_cont_full_st” and the time period of maximum number of occupancy of entities is refreshed in variable “place_cont_full_ts”. If the total number of parts in the buffer is lower than the buffer´s limitation, function *“fce_Priority_previous”* tries to find an operation for the previous resource. Function *“fce_Unpack_previous”* checks if entities are a product of an unpacking operation.

```
IF place_cont_full_st (part_number,crate,router,vol) >
place_cont_full_mx (part_number,crate,router,vol)
  place_cont_full_ts (part_number,crate,router,vol) = 0
ENDIF

IF place_cont_full_st (part_number,crate,router,vol) + 1 >=
place_cont_full_mx (part_number,crate,router,vol)
  place_cont_full_mx (part_number,crate,router,vol) =
place_cont_full_st (part_number,crate,router,vol) + 1
  place_cont_full_ta (part_number,crate,router,vol) = TIME
ENDIF

IF fce_Unpack_next (routing_list
(router),ELEMENT,part_number,crate,router,vol,operation_list (router)) =
1
  place_cont_full_st (part_number,crate,router,vol) =
place_cont_full_st (part_number,crate,router,vol) + 1
ENDIF
fce_Priority (part_number,crate,router,vol,ELEMENT)
```

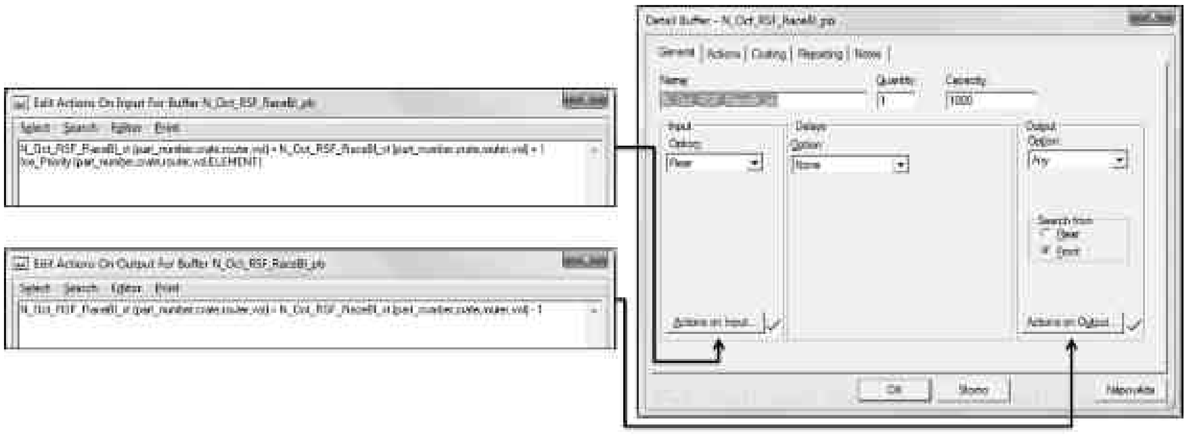
Figure 73 - Buffer: Action on Minimum delay code

```
place_cont_full_st (part_number,crate,router,vol) = place_cont_full_st
(part_number,crate,router,vol) - 1
IF place_cont_full_st (part_number,crate,router,vol) =
place_cont_full_mx (part_number,crate,router,vol) - 1
  place_cont_full_ts (part_number,crate,router,vol) =
place_cont_full_ts (part_number,crate,router,vol) + TIME -
place_cont_full_ta (part_number,crate,router,vol)
ENDIF
IF NPARTS (place_cont_full_bu) < INAMEVAL (place_cont_full_li)
  fce_Priority_previous (part_number,crate,router,vol,ELEMENT)
ENDIF
fce_Unpack_previous (routing_list
(router),ELEMENT,part_number,crate,router,vol,operation_list (router))
IF boo_Dyn_write = 1
  fce_Dynamic_array (ELEMENT,TIME,part_number,crate,router,vol,2)
ENDIF
```

Figure 74 - Buffer: Action On Output code

d) <u>Input System Buffer</u>

Prebuffer (internal IDS name) is a specific buffer that stores entities before their entrance to the system. They can represent storages with raw material or semi-finished products. It is important to hold some amount of entities for their loading into the first resource. *Prebuffer* doesn´t involve any detailed statistics; its function is virtual space. Internal logic is different from standard buffer; it contains only the increasing/decreasing number of entities in the buffer as shown in Figure 75.


Edit Actions On Input For Buffer N_Oct_RSF_RaceBl_pb

```
N_Oct_RSF_RaceBl_st [part_number.route.router.val] = N_Oct_RSF_RaceBl_st [part_number.route.router.val] + 1
Ice_Priority [part_number.route.router.val].ELEMENT)
```

Edit Actions On Output For Buffer N_Oct_RSF_RaceBl_pb

```
N_Oct_RSF_RaceBl_st [part_number.route.router.val] = N_Oct_RSF_RaceBl_st [part_number.route.router.val] - 1
```

Detail Buffer - N_Oct_RSF_RaceBl_pb

Name: [illegible] Quantity: 1 Capacity: 1000

Input Option: Rear

Delays Option: None

Output Option: Any

Search from: Rear [✓ Front]

Figure 75 - Details of input system buffer

4.4.5 Operation properties

In the IDS, operation is a configuration of variables and parameters that describes target activities processed in the resource.

Each operation has three phases, **Inputting**, **Processing** and **Outputting**. In the first phase, input sequence of entities is prepared. After that, the process starts by running the operation time and, after the expiration, the output sequence of entities is outgoing from the resource, as final products of operation.

In the WITNESS Simulation tool, there is no default object with properties of operation, thus in IDS, it is done by a set of arrays.

Definition of operation

The name of the operation variables is based on its name and suffix. For example, the name of an array for input information of operation "Take_down" is "Take_down_io". Table 18 involves all the required information for a full description of the operation. Each array consists of columns with specific values; each row of array is for different inputting/outgoing entities. It is used in the assembly kind of operation, dividing operation or, for instance, injection operation processed on moulding machines where several different parts can be produced from a form by one cycle machine. In those cases, the number of rows also gives information about order of inputting/outgoing entities. Values in these arrays are assigned to entity attributes values.

By these approaches, it is possible to describe any kind of operation and also to use it in automatic generated models and easy running of experiments. All the required arrays are automatically generated as well as values that are assigned in the initialisation phase of the simulation.

Table 18 – Variables and properties of operation

Variable	**Meaning**	**Declaration type**
[Operation name][**_io]**	Inputting sequence definition(part_number, crate, router, vol, amount in)	Integer
[Operation name][**_oo]**	Outputing sequence definition(part_number,crate, routing, amount out, colour, icon, vol, stochastics)	Integer
[Resource name][**_tm**]	operation time	String (expression)
[Operation name] [**_pn**]	name of outgoing entities corresponding to the part_number in the [Operation name][**_oo]**	String

The operation time can be given as a real number, by a stochastic function (e.g. by Normal distribution function) or by an expression using internal variables. For example, the operation for loading entities, depending on the inputting amount, can be "10 * [Resource name][_in]" where 10 means the loading time for one entity and [Resource name][_in] is the variable for input quantity of [Resource name]. Operation time based on the number of a passed operation can be based on the values from the array "([Resource name][_tt] ([Resource name][_pa],3))".

4.4.6 Other objects in the simulation model

a) Human resource object properties

Human resources belong to the basic system objects. In IDS, they have a simple use and control logic. Defined human resource is generated and assigned to the resources according to the DB (see Figure 55). The operation can only be processed if human resource is available.

b) Output system resource

A similar function as *"Input system buffer"* is "*Output system resource*". It is a special kind of resource that pulls full-processed entities from buffers and pushes them to the "Ship". The term "Ship" in the WITNESS simulation tool means "out of the simulation model"; entities cross frontiers of the described system and then disappear. Information about that is included in the table "Output" in the property *"output_from the system"*.

c) Stochastic functions in simulation

Stochastic functions are very important issues of simulation. With these functions, it is possible to study issues of randomness in a system and its influences.

In IDS, there are several kinds of objects properties that can be described by stochastic functions:

- operation times (production), service, breakdown etc.
- material flows of entities in the systems
- properties of entities (e.g. a quality)

Operation times can be defined by WITNESS stochastic function as "NORMAL" distribution function; service and breakdowns can be defined by log-normal distribution ("LOGNORML"), Negative Exponential distribution ("NEGEXP"), gamma distribution ("GAMMA"), ERLANG K distribution ("ERLANG"), etc.

Material flows can be established by the behaviour of other objects, resources, etc., and by the definition of external logic control.

Output quality (good products or scraps) can be defined by the attribute *"vol"* that can get integer value by statistic function e.g. *"IUNIFORM (1,5)"* or by user defined profiles. The example of profiles use is in Section A.4.3.

4.4.7 The initialisation of model

The Initialisation phase processes the setup of simulation objects and prepares the model for a simulation running. Data are loaded into the model from DB and assigned to arrays and variables. After that, objects are configured. The full programme code is included in Appendix (Section A.3.3.3).

Setup of initialisation

The Initialisation phase can be set up by several parameters. Those are mentioned in Table 19.

Table 19 – Initialisation parameters

Parameter	Meaning
booOK_show_message	Show control OK messages
booWrite	Write details into files
boo_Dyn_write	Write(1) buffer values (dynamic)
booPriority	0 = searching the optimal, 1 = search the first possible

```
!code for establishing size of array ListOfMachines

OK = SETINFO ("DETAIL\nSELECT\nListOfMachines\nNAME OF VARIABLE:
ListOfMachines;\nQUANTITY: 1," + number_machines + ";\nREPORTING: Yes;\nEND
ListOfMachines\nEND SELECT\nEND DETAIL\n")
x = 0

!SQL query - list of machines:

sql_general = "SELECT ListOfInventory.name_machine_sim_inv FROM
ListOfInventory ORDER BY ListOfInventory.id_inventory;"

!assigning values:

db_machines.OpenRecords ()
WHILE db_machines.IsAtEnd () = 0
      x = x + 1
      db_machines.LoadCurrent ()
      ListOfMachines (1,x) = db_machines.name_machine_sim_inv
      db_machines.NextRecord ()
ENDWHILE
db_machines.CloseRecords ()
```

Figure 76 – Function Array

Function Arrays()

This function involves codes for array sizing and assigning values into arrays. The full code is in Appendix (Section A.3.3.3.). Example of code (Figure 76) shows the processing of array: establishing size, loading and assigning data from database.

Function Details()

The setup of some objects details are processed in function *function_Detail()*. There are Details of Parts; Details of Buffers; Definition of Part Arrivals by part files; Profiles definition, etc. Codes of this function can be directly involved in the Initialisation section; it is separated due to size limitation of WITNESS text space for code (64kb). Example from this function is in Figure 77 - establishing machine (resource) details as input quantity, output quantity, cycle time, etc.

```
!+++++++++++++++++++++++++DETAILS MACHINES:++++++++++++++++:
IF intRedef = 1
      x = 0
      strdef = ""
      FOR x = 1 TO number_machines
            strdef = "DETAIL\nSELECT\n" + ListOfMachines (1,x) + "_pr\n"
            strdef = strdef + "TYPE: General;\n"
            strdef = strdef + " * Input qty:" + ListOfMachines (1,x) +
"_in;\n"
            strdef = strdef + " * Output qty:" + ListOfMachines (1,x) +
"_ot;\n"
            strdef = strdef + "* Inherit Attribute Values: Yes;\n"
            strdef = strdef + "CYCLE TIME:" + ListOfMachines (1,x) + "_tm
;\n"
            strdef = strdef + "ACTIONS, Start\nAdd\n"
            strdef = strdef + "!Variable reseting \n"
            strdef = strdef + ListOfMachines (1,x) + "_hp=1\n"
            strdef = strdef + ListOfMachines (1,x) + "_oc=0\n"
            strdef = strdef + ListOfMachines (1,x) + "_ic=0\n"
            strdef = strdef +"OK = RVARSET ("+ ListOfMachines (1,x)
+"_tt,TIME,"+ ListOfMachines (1,x) +"_pa,"+ ListOfMachines (1,x)
+"_pa,1,1)\n"
            strdef = strdef + "End Actions\n"
            strdef = strdef + "Output_From: Any;\n"

            strdef = strdef + "\nEND" + ListOfMachines (1,x) + "_pr\n"
            strdef = strdef + "END SELECT\nEND DETAIL\n"
            OK = SETINFO (strdef)
      NEXT
ENDIF
```

Figure 77 – Function Detail

Rule On Input

It is not worthwhile to describe the full code of Initialisation. "Rule on Input" (Figure 78) was chosen as an example. The rest can be seen in Appendix (Section A.3.3.3).

```
!++++++++++++++++++++RULE ON INPUT++++++++++++++++++++++++++++
IF intRedef = 1
x = 0
y = 0
z = 0
ww = 0
strdef = ""
FOR x = 1 TO number_machines

      IF ListOfKindOfProcess(1,x)<>"transport_helper" AND
      ListOfKindOfProcess(1,x)<>"milk_run" THEN
             strdef = "DETAIL\nSELECT\n" + ListOfMachines (1,x) + "_pr\n"
             strdef = strdef + "INPUT RULE:"
             strdef = strdef + "IF " + ListOfMachines (1,x)+"_pc<>0\n"
            strdef = strdef + "MATCH/CONDITION (part_number = INAMEVAL
            (SNAMEVAL ("+ListOfMachines (1,x)+"_ow),"+ListOfMachines
            (1,x)+"_hp,1) AND router = INAMEVAL (SNAMEVAL ("+ListOfMachines
            (1,x)+"_ow),"+ListOfMachines (1,x)+"_hp,3) - 1 AND crate =
            INAMEVAL (SNAMEVAL ("+ListOfMachines
            (1,x)+"_ow),"+ListOfMachines (1,x)+"_hp,2) AND vol = INAMEVAL
            (SNAMEVAL ("+ListOfMachines (1,x)+"_ow),"+ListOfMachines
            (1,x)+"_hp,5)) STR2NAME ("+ListOfMachines (1,x)+"_wh
            (1,"+ListOfMachines (1,x)+"_hp)) #(1)\n"

            strdef = strdef +"ELSE\n"
            strdef = strdef +"Wait\n"
            strdef = strdef +"ENDIF\n"
                  IF booWrite=1 THEN
                        WRITE write_in_rule_pr strdef
                  ENDIF
            OK = SETINFO (strdef)
            strdef = ""
      ENDIF
NEXT !x

      IF booWrite=1 THEN
      CLOSE (write_in_rule_pr)
      ENDIF
ENDIF
```

Figure 78 – Rule on Input example

4.4.8 Principle of external logic control

Logic control functions manage queues of events. Also, with logic control functions, it is possible to control material flows of entities, selection operation, etc. WITNESS simulation tool provides an extensive set of functions and rules for it. The most used rules are *"Pull"* for pulling entities from another resource or buffer, *"Push"* for pushing entities to another resource, *"Sequence"* to define sequence of inputting entities, *"Match"* for selection entities based on attributes or parameters, and others (*"MOST"*, *"LEAST"*, *"PERCENT"*, *"WAIT"*).

In IDS, there is an effort to describe control logic by the most general way so that is possible to use in many different tasks and projects. In this way, control logic is done by a combination of *"IF"* conditions and a set of functions for the selection operation for the given resource.

In Figure 79, there is a schema of logic control used in IDS. Solid lines and arrows represent movements of entities based on external control logic, and the dotted lines represent internal logic in the resource and buffers, as described in sections above.

Entity can move from "Buffer_1" to "Resource_A" or to "Resource_B". A set of operations is defined for given resources and entities. The selection of the resource (operation) is based on function *"fce_Priority"* that is triggered in the event of an entity entering "Buffer_1" or the required minimum delay time passes. After that, an operation is selected and entity is sent to the determinate resource. After processing the operation, this entity enters "Buffer_2" and it triggers the function that searches another possible operation for the entity itself *("fce_Priority()").* It is also triggered a function *("fce_Priority_previous")* for searching an operation for previous resources from where an entity could have come ("Resource_A" or to "Resource_B"). The entity is waiting for the next operation and, by the time it leaves "Buffer_2", the function *("fce_Priority_previous")* is triggered again. This is mainly for some cases, if the capacity of "Buffer_2" is limited and resources were blocked.

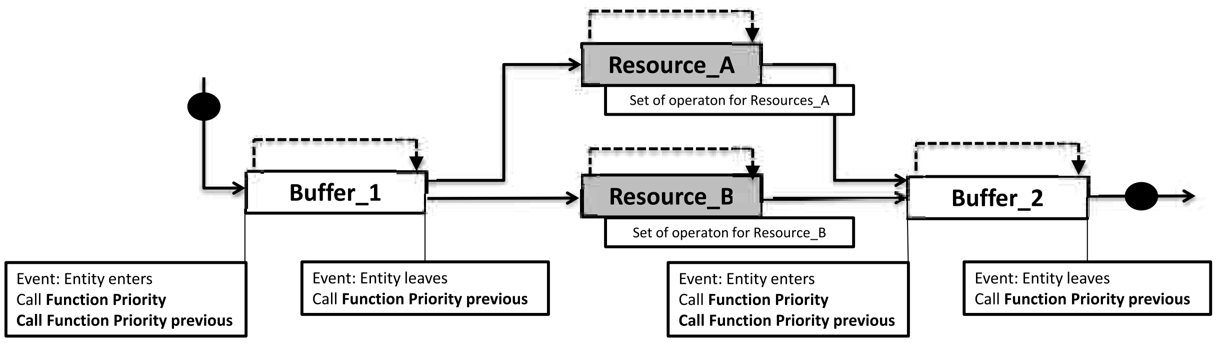

Resource_A
Set of operaton for Resources_A
Buffer_1
Resource_B
Set of operaton for Resource_B
Buffer_2
Event: Entity enters
Call **Function Priority**
Call Function Priority previous
Event: Entity leaves
Call **Function Priority previous**
Event: Entity enters
Call **Function Priority**
Call Function Priority previous
Event: Entity leaves
Call **Function Priority previous**

Figure 79 - Schema of control logic – internal and external

"Fce_Priority" tries to push entities to the next resource and function *"fce_Priority_previous"* tries to pull entities from buffers. Both functions involve codes for searching operation. It is based on the following rules and conditions:

- priorities of operation
- current state of processes – only "idle"
- inputting parts available
- available space for outgoing parts

Priorities of operations are defined in the table *"Tbl_working_possible"*. The operation with the highest priority is selected. In case of tie, the operation with the lowest position in the table is selected. The resource for the selected operation must be in **idle state** and all inputting entities must be available in the input buffers. These conditions couldn't stand if the operation property **waiting for all entities** is allowed (function "*fce_part_wait*", see Section A.4.2).

If required, there must be also a valid condition of avoiding blocked resources. It means that there must be available space in the output buffer for outgoing entities, as described in Section A.4.2.

A shorter code of the *"Fce_Priority"* is in Figure 80 and a full code is in Appendix (Section A.3.3.3), as well as *"fce_Priority_previous"* code.

The output of these functions is a complete setup of operation variables for a given resource: operation name, operation ID number and names of buffers from where entities are pulled to resources.

```
!searching in the table tbl_WITNESS_input_int for possible operations,
based on part_id, crate_id, route_int+1, vol_int of the incoming entity:
FOR i_i = 1 TO number_WITNESS_table_rec

IF (tbl_WITNESS_input_int (1,i_i) = part_id) AND (w_int (2,i_i) = crate_id)
AND (tbl_WITNESS_input_int (3,i_i) = route_int + 1) AND
(tbl_WITNESS_input_int (4,i_i) = vol_int)

strHelp_pa = tbl_WITNESS_input_str (2,i_i) + "_pa"


! check of the highest priority and idle of resource:
IF (tbl_WITNESS_input_int (5,i_i) >= priority_basic) AND (INAMEVAL
(strHelp_pa) = 0)

strHelp_operation = tbl_WITNESS_input_str (1,i_i) + "_io"
!cycle for all inputs of the operation; j_j=operation rows index
FOR j_j = 1 TO maxdim (strHelp_operation,1)
!arrays of storing information
strMachine_zi=tbl_WITNESS_input_str (2,i_i)+"_zi"
strMachine_zs=tbl_WITNESS_input_str (2,i_i)+"_zs"

FOR m_m=1 to maxdim(strMachine_zs,2) !

      IF [comparing the inputs entity attributes (part, crate,router, vol)
      in the storing information (array strMachine_zi)]

      name_storage=SNAMEVAL(strMachine_zs,1,m_m) !name of storage
      strHelp_st=LEFTSTR (name_storage,STRLEN (name_storage) - 3)+"_st"

      IF (fce_part_wait(strHelp_st,tbl_WITNESS_input_int (6,i_i),j_j)=1)
      AND (fce_Blocation (tbl_WITNESS_input_str
      (2,i_i),tbl_WITNESS_input_int (6,i_i)) = 1)

            strHelp_buffer = SNAMEVAL(strMachine_zs,1,m_m)
            Array_where (1,j_j) = strHelp_buffer
            str_aktual_machine_ow=tbl_WITNESS_input_str (2,i_i) + "_ow"
            real_ow_machine= tbl_WITNESS_input_str (1,i_i)!name operation
            real_pc_machine = tbl_WITNESS_input_str (2,i_i) + "_pc"
            real_pc_ID_oper = tbl_WITNESS_input_int (6,i_i)
            priority_basic = tbl_WITNESS_input_int (5,i_i)
!end of searching, if priorities are not required…
            IF booPriority=1 AND maxdim (strHelp_operation,1)=1
                  i_i=number_WITNESS_table_rec
            ENDIF !priority

            m_m=maxdim(strMachine_zs,2)

      ENDIF
      ENDIF
NEXT m_m
NEXT j_j
ENDIF !priority_basic
ENDIF
NEXT i_i
!-------assigning of operation variables:-------------
IF real_pc_ID_oper<>0

      name_operation = ListOfOperations (1,real_pc_ID_oper)+"_io"
      OK = SVARSET(str_aktual_machine_ow,name_operation) !_ow
      OK = IVARSET (real_pc_machine,real_pc_ID_oper) !_pc

FOR k_k = 1 TO maxdim (name_operation,1)
      strhelp_kam = LEFTSTR (real_pc_machine,STRLEN (real_pc_machine) - 3)+
      "_wh"
      SVARSET (strhelp_kam,Array_where (1,k_k),1,1,k_k,k_k)
NEXT
ENDIF
```

Figure 80 - “Function Priority” code

4.4.9 Transportation functions

Transportation function, involved in simulation model, is a helpful tool for design transportation systems. Implementation is described in Section 4.7, practical examples in Section A.4.10 and A.4.11.

Function_transport_time()

This function calculates the transportation time required for movement between the actual position of transporter and the start position (*Fce_transport_time_tp)* or between the start and the target position (*Fce_transport_time)* (see Figure 81). Input data are ID_operation *(“ID_oper”*), ID_resource (*“ID_res”*), the transporter name (*“name_tp”)* and the actual position (*“act_position”*). Transportation time is calculated by the distance divided by the speed of transporter. The distance is found in the array *from_to_value()* containing the distance for all possible ways. Simulation model involves the arrays concerning transportation - array *ID_oper_from_to()* with ID numbers of transportation operations and an array *“Oper_from_to()”* that involves names of stations “from” (loading) and “to” (target station) for a given operation. Distances between each station are in the arrays *“from_to_str()”* and *“From_to_value”*.

```
!Speed:
tbl_speed (1, ID_res)= speed
if tbl_speed (1, ID_res)=0
      speed =1
endif

FOR i_i = 1 TO maxdim (ID_oper_from_to,2)
      IF ID_oper_from_to (1,i_i) = ID_oper
            from_1 = SNAMEVAL (act_position) !actual position
            start_1 = oper_from_to (1,i_i) !start_position
            i_i = maxdim (ID_oper_from_to,2)
      ENDIF
NEXT
!searching for distance:
FOR j_j = 1 TO maxdim (from_to_str,2)
      IF from_to_str (2,j_j) = start_1 AND from_to_str (1,j_j) = from_1

      transport_time = from_to_value (1,j_j) / (speed)

      j_j = maxdim (from_to_str,2)
      ENDIF
NEXT
RETURN transport_time
```

Figure 81 - Transportation function code

4.5 CAD systems

In this section, the principle of using CAD system in IDS is described, as well as important functions that are helpful in facility layout design. Basically, the AutoCAD objects used are **block**, **block attribute**, **polyline**, **layers** and **groups**.

As established in the challenges, the main functions of CAD system integrated in IDS are the layout design and the spatial solution of objects in facilities. IDS supports two types of layouts that are based on the design phase and on the level of data level: Factory layout design and Facility layout design.

Factory layout design consists of general factory design, placement of departments, established department sizes, etc. In IDS, several methods can be used for helping designed optimal layouts - CRAFT, CORELAP, Graph Theory Block Layouts placement method. These optimization algorithms are based on material flows and transportation costs. Material flows are automatically generated into the layout to easily see products flows and their movements in the facilities.

Facility layout design in IDS is supported by basic layout patterns generation. Basic patterns are production lines, production cell and process layout (job-shop), or pattern with resources in a row, a column or a diagonal. The IDS functions also help to design buffers and their sizes. It is supported transportation network design and finding the shortest path.

4.5.1 General CAD

As the CAD system integrated in IDS, the AutoCAD application was chosen. Codes for functions and input/output forms are written in VBA macros and with the use of some ActiveX objects and external libraries, e.g. *"AutoCAD DWGthumbnails Conrol"* for displaying previews of drawings, *"Microsoft TreeView Control 6.0"* for displaying tree structure of CAD blocks library, etc.

The complete list of controls, libraries and references is in Appendix (Section A.3.3) as well as a programme code, saved in AutoCAD project format (*"dvb"* file).

4.5.2 AutoCAD objects

For the IDS purpose, AutoCAD and graphic objects are used, as block, block attribute, dynamic block, group, layer and polyline (Finkelstein, 2006). These objects and their properties are used in the layout design functions. For example, blocks are used as resource drawings or polylines for displaying material flows.
The following sections contain a description of these objects, examples of VBA codes and principle of use.

a) <u>Block and attribute</u>

Block is a group of objects fixed together. A block creates one single object thus it is easy to move, scale or rotate with all the involved objects. It is used in cases where the same group of objects is placed several times in the drawing. Each of these inserted instances is referenced to the original block definition. Thus, if it is changed, all instances are automatically updated. This block definition is stored in the drawing file only once. The inserted instances are copies of it. Blocks of drawings can be saved externally, creating **libraries**. It is possible to use it in many drawings or share between users. Examples of blocks are in Figure 82.

Attributes are text labels associated with blocks. Each block instances can have different values of attributes, thus it is possible to distinguish them by each attribute.

A **dynamic block** contains certain properties as parameters, so this type of block can be changed by them - mainly positions, rotations, visibility of inner blocks objects. It gives great flexibility for a work with blocks.

Blocks are frequently used in IDS for their features. It is possible to generate layouts automatically, save time and make the project more uniform. Basic operations with blocks used in automatic generation functions are inserting, adding attributes, loading information from block instances and deleting blocks. Those operations with blocks are used in the generation of layouts patterns (see Section see A.4.8), factory layouts (see Section A.4.4) or buffer sizing (see Section 4.7.3). A library of blocks is built for easy use of them (see Section A.4.1).

In the IDS projects, drawing blocks are used for graphic representation of resources, buffers and parts. Blocks contain geometrical information as coordinates, rotation, scale, etc. These blocks must also involve a set of **attributes** for the determination of their name and other properties. In the case of resource or buffer blocks, it is just a name of objects; blocks representing entities involve more attributes as entity's name, actual crate in which an entity is packed, buffer's name (current storage's name), variant of entity and unpacking information (carrier) (see Table 20).

Each resource and buffer must also involve **geometrical points** representing start and end points of material flows. The unique names of these points are important for the determination of coordinates. These points are represented by simple blocks with specific names based on the resource or the buffer's name and given suffix. For resource, point blocks' names have to end on [_pr_in] and [_pr_out] and for buffers on [_bu_in] and [_bu_out]. In Figure 82, blocks representing geometrical points are a small circle with arrows that symbolizes material flow direction. These points are automatically generated in the process of inserting blocks into the layout.

Table 20 – Block attributes and input/output points

Object name	**Input/output geometrical points**	**Attributes**
Resource blocks	[_pr_in] [_pr_out]	NAME_PROCESS
Buffer blocks	[_bu_in] [_bu_out]	NAME_BUFFER
Part blocks	Not involved	PART_NAME CRATE_NAME BUFFER_NAME VARIANT CARRIER_FOR_UNPACK

Process_Buffer **block**

It is a special type of block containing process and buffer attributes together, with only one input/output geometrical point. It is used in cases of simple factory layouts, where buffers are not considered and when there are no operation data available. An example is in Figure 201.
As mentioned above, several basic procedures with blocks are used in IDS (inserting, adding attributes, loading information from block instances and deleting blocks). As example of the work, inserting blocks procedure is chosen; the rest is possible find in Appendix (Section A.3.3).

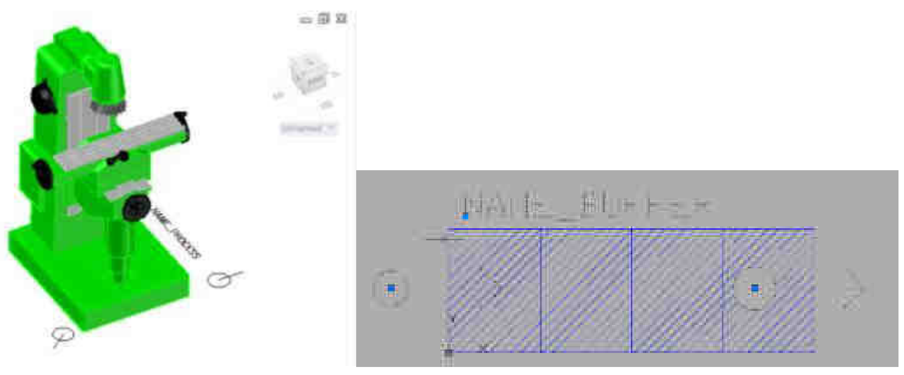


Figure 82 - Examples of blocks: 3D block (mill machine) and schematic buffer block

Inserting blocks and assigning attributes

Procedure for inserting blocks is in Figure 83. Code involves also assigning value to attribute.

```
Dim pointStart(2) As Double
Dim xref As AcadBlockReference
Dim arrBlockAtr As Variant
…
strNameBlock = arrBuffers(i)
…
pointStart(0) = X_coordinates
pointStart(1) = Y_coordinates
pointStart(2) = Z_coordinates

Set xref = ThisDrawing.ModelSpace.InsertBlock(pointStart, strNameBlock, 1,
1, 1, 0)
xref.layer = "Layout_buffers"
arrBlockAtr = xref.GetAttributes 'get attributes

For count = LBound(arrBlockAtr) To UBound(arrBlockAtr)
      If pole(count).TagString = "NAME_BUFFER" Then
      arrBlockAtr (count).TextString = arrActiveParts(i_i) & "_buffer"
      Exit For
      End If
Next
```

Figure 83 - Inserting blocks procedure

b) Group

Another possibility of putting objects together is a set of selected objects. Besides blocks, objects can be put together in Groups, where they can be edited directly. Groups are used in material flows displaying, as a way to assign a unique name into polyline that is not possible in the current version of AutoCAD. Material flows polylines can be joined to blocks with a unique name and information to attributes at the cost of losing the possibility of direct polyline editing. In order to edit polylines, e.g. creating specific flow shape, the block must be exploded, edited and then saved again. This approach is used e.g. in the application MatFlow.

Another possibility is to add a text legend with flow information next to the polyline. This approach is used in the application FastDesign. It is user-friendly but there can be a problem if the flow is moved far from the legend that stayed in the same position.

c) <u>Layer</u>

A layer is a powerful tool for drawing an organisation of objects. It supports distinguished objects by different colours, linetypes and lineweights. It alows controlling visibility, transparency and locking layers so that objects on that layer cannot be edited. Working with layers is simple, as shown in the code in Figure 84.

```
Dim strTempName As String
Dim oLayer As AcadLayer

Dim color As AcadAcCmColor
Set color = AcadApplication.GetInterfaceObject("AutoCAD.AcCmColor.18")

(arrays strName_parts(i_i), intLayer_color_R(i_i), intLayer_color_G(i_i),
intLayer_color_B(i_i)) contain information about layer name and RGB color
definition)

strTempName = strName_parts(i_i) + "_ideal"
Set oLayer = ThisDrawing.Layers.Add(strTempName)
Call color.SetRGB(intLayer_color_R(i_i), intLayer_color_G(i_i),
intLayer_color_B(i_i))
oLayer.TrueColor = color
…
```

Figure 84 - Layer procedure

d) <u>Polyline</u>

Polylines are single objects built from line segments and arcs. Polylines consist of a set of vertices creating connected segments. Polylines can have different width of each segment. In IDS, polylines are used for displaying material flows. The basic operations are **polyline drawing** and **saving information** that represents material flows.

Polyline drawing procedure

The following code generates polyline based on start and final points; their coordinates are loaded from the DB. Figure 85 contains short-cut code; the full code can be found in Appendix (Section A.3.3).

Polyline saving procedure

This code consists of loading a list of flows from DB, finding group with a name containing flow name and then getting information about coordinates of vertex, layer, width, etc. It is saved in DB. Full code is in Appendix (Section A.3.3).

```
'Procedure: Connection_DB
'AutoCAD objects declaration:

      Dim objEnt(0), objEnt_1(0) As AcadPolyline
      Dim From_point_x,  From_point_y,  From_point_z As Double
      Dim To_point_x,  To_point_y , To_point_z As Double
      Dim new_point As Integer
      Dim arrCoordinates() As Double
      Dim objGroup As AcadGroup'Group
'Calculation of the maximum polyline width:
      boo_weight = "frequency"
      Koeficinet_width_polyline (boo_weight)
'SQL query searches for material flows definition:
strQuery = "SELECT Tbl_flows_ideal.from_process_point,
Tbl_flows_ideal.to_process_point,Tbl_flows_ideal.part_name_prepare,...Tbl_w
eight.weight, [number]*[weight] AS value_flow, Tbl_weight.cost_p_unit FROM
Tbl_weight INNER JOIN Tbl_flows_ideal ON ...;"

oRecordset.Open strQuery, oAccess, adOpenKeyset, adCmdText
With oRecordset
      Do While Not .EOF
            From_point_x = 0; From_point_y = 0; From_point_z = 0
            To_point_x = 0; To_point_y = 0; To_point_z = 0
'Procedure: Finding coordinates for given Input/output points
(!from_process_point and ! to_process_point), SQL query searches in the
CAD_basic_points_coordinates table and returns x,y,z coordinates

'adding coorinates of points to the array arrCoordinates:
      new_point = 0
      arrCoordinates(new_point) = From_point_x
      new_point = new_point + 1
      arrCoordinates(new_point) = From_point_y
      new_point = new_point + 1
      '…similarly for the rest of points: From_point_z,
      …To_point_x,To_point_y, To_point_z
'drawing of polyline:
      Set objEnt(0) = ThisDrawing.ModelSpace.AddPolyline(arrCoordinates)
'width of polyline:
      Number_flow = !Number
      objEnt(0).ConstantWidth = Number_flow * dblKoeficientWidthFlow
'Adding polyline to the specific group:
      strName = !CAD_group_name
      Set objGroup = ThisDrawing.Groups.Item(strName)
      Set objGroup = ThisDrawing.Groups.Add(strName)
      objGroup.AppendItems objEnt
'Layer of polyline:
      objEnt(0).Layer = !part_name_prepare + "_ideal"
.MoveNext
Loop
.Close
End With
```

Figure 85 - Polyline drawing procedure

4.5.3 Material flows display

Flows and relationships displayed in the layout are very helpful to make easy the understanding of products movements, intensities of transportation and relationships in production system. Graphic displaying of material flows can help to discover potential places with enormous traffic, non-straight flows, very intensive flows for very long distances and other problems. Flows can be taken as a parameter for the layout alternative evaluation or as input data for the layout optimization – e.g. CRAFT method.

The material flow is specified by the following information: start position, end position (direction), intensity, entity, kind of crate and shape of the flow. In Table 21, there is a summary of these properties. All these information are given by data from the DB, except the shape that is designed in the CAD system. These properties help to distinguish different kind of flows as mentioned in **Table 12**.

Automatic displaying of material flows is based on the procedures described in Section 4.5.2.

Table 21 - List of flow properties

Information	Representation	Notice
Specific part or crate	Layer	Layer filtering, Use of different colours
Shape of flow	Line/Polyline	
Intensity	Width	Exact value of flow in Group name, link reference, in DB, recalculation (by the maximum polyline width)
Direction	Polyline properties Reference description	
Full flow description	Reference	Start and end name of resource, product name, crate name, order in operation sequence, variable, real intensity
Kind of flow	Group assigning	Filtering by VBA macros

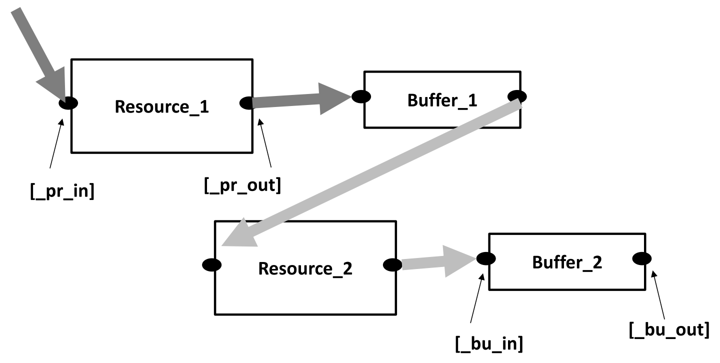


Figure 86 - Schema of material flows between resources

Start and end points

Flows can be divided into incoming to the resource and outgoing from the process. Flows inside the resource or the buffer are not taken into account (see Figure 86).

Each resource and buffer involves geometrical points representing start and target point of material flow. These points are automatically generated in the process of inserting blocks into the layout.

Flows intensity

The flow intensity is represented by the **width** of the polyline. The real values of the flows are recalculated based on the maximum polyline width established by the user (*maxFlow_width*). The reason of it is to avoid displaying flows with enormous widths of polylines. Therefore, generated flows in the layout don´t have their actual width but modified by the coefficient *dblKoeficientWidthFlow*. So, widths of flows in the layout are in relative rate to each other. This coefficient is found by the procedure in Figure 87.

```
maxFlow_width=200

strQuery = "SELECT Max(Tbl_flows_ideal.number) AS MaxOfnumber FROM
Tbl_flows_ideal;"

      oRecordset.Open strQuery, oAccess, adOpenKeyset, adCmdText
      With oRecordset
      maxFlow_real = !MaxOfnumber
      .Close
      End With

If maxFlow_real <> 0 Then
      dblKoeficientWidthFlow = maxFlow_width / (maxFlow_real)
Else
      dblKoeficientWidthFlow = 0
End If
```

Figure 87 - Schema of material flows between resources

Flows information

In AutoCAD, it is not possible to link extra information, such as exact value of flow or name, directly to the polyline objects. Each polyline has to be uniquely determined to load information into the DB. Due to these reasons, each polyline is assigned to a specific *group* and information is held in the group´s unique name. The name of the group consists of the following information: *part_name , crate_name ,router , vol* (attributes of entity) and *flow intensity*.

a) Work with flows

In IDS, work with flows has the following steps:

- Preparing data for flows
- Automatic generation of flows by VBA
- Manual arrangement of polyline shape
- Saving coordinates and polyline information into the DB
- Analysis of flows

Preparing data for flows

The required input data for automatic generation of flows are coordinates of start ("*from*") and target ("*to*") points, intensity of flows and kind of flow.

There are two options for data preparation. Firstly, the user can do it manually. The user fills appropriate tables with from-to stations and intensity. These values can be based on estimation and in cases where there is no exact data. The second option is to use the results data from the simulation model considering stochastic influences of processes as waiting, fails, blockage, etc. It is a complete data set involving a number of throughput entities and their attributes (part, crate, routing, variable). Both data sets can be used for **optimization** algorithms based on material flows as in CRAFT.

Automatic generation of flows by VBA

Material flows are automatically generated by VBA macros using the code *"Polyline drawing"* (in Section 4.5.2)

Based on the shape, flows can be drawn as a straight line with only start and target point or as a curved polyline with more vertices and segments. In this case, this line is divided into 10 segments by default and the user arranges manually the polyline into the required shape just by moving vertices.

Manual arrangement of polyline shape

Generated polylines don´t consider limitations of buildings, aisles, etc. It is necessary to arrange them according to these constraints; it can be done manually by the user as in the figures in Figure 228). Another option is the automatic generation of material flows based on the prepared transportation network where realistic shapes are considered (see Section 4.7.3).

Saving coordinates and polyline information into the DB

Material flows are saved in DB based on the procedure *"Polyline saving procedure"* mentioned in section 4.5.2.

Analysis of flows

Material flows are then analysed in DB by built-in functions - see example in Figure 205.

4.5.4 CRAFT

Heuristics of CRAFT algorithm were described in Section 2.2.1. Briefly, the CRAFT method optimises the placement of departments in the layout based on material flows, distances between department centroids and costs. The aim is to achieve the minimal total transportation costs.

The following section explains the development of CRAFT. It is based on the work of Professor Jensen from The University of Texas (Jensen & Jonathan F., 2003) and on the code presented on the web page (www.me.utexas.edu/~jensen/ORMM/, 2011) and then it was adjusted to IDS. The codes in following sections are simplified; some not important orders and sections are omitted. The complete codes are included in Appendix (Section A.3.3).

CRAFT development has the following parts:

- Initial layout
- Optimization
- Drawing an optimised layout

As mentioned in the heuristics, the CRAFT method needs an initial solution of the layout that CRAFT tries to improve by optimization. The last part of the code builds a layout by inserting blocks that represent departments.

a) <u>Drawing the initial layout</u>

The initial layout can be manually drawn by a user or it can be automatically generated by a set of IDS functions. Among the input, such as department information (size) and material flows between them, a *sequence* of departments is needed. Based on this sequence, departments are placed into the layout. This sequence of departments can be done by a user or randomly generated for the initial layout. This principle is used in CRAFT modification, called *microCRAFT* or *mCRAFT*.

A solution for generating the initial layout is based on filling layout space by full SFC (*Space Filling Curves*, see Section 2.2.1). For the purpose of IDS, it is simplified. In Figure 88, there is a schema of filling factory space given by width and length: the leading curve goes through the factory space, filling all the department width until it reaches the total factory length. The process is repeated in the rest of factory space until all departments are placed.

The department has a defined size which is equal to the appropriated number of square units. During the generation, the layout is represented by 2D integer array where elements contain indexes of departments. In the last phase, these elements are replaced by the drawing blocks into the layout.

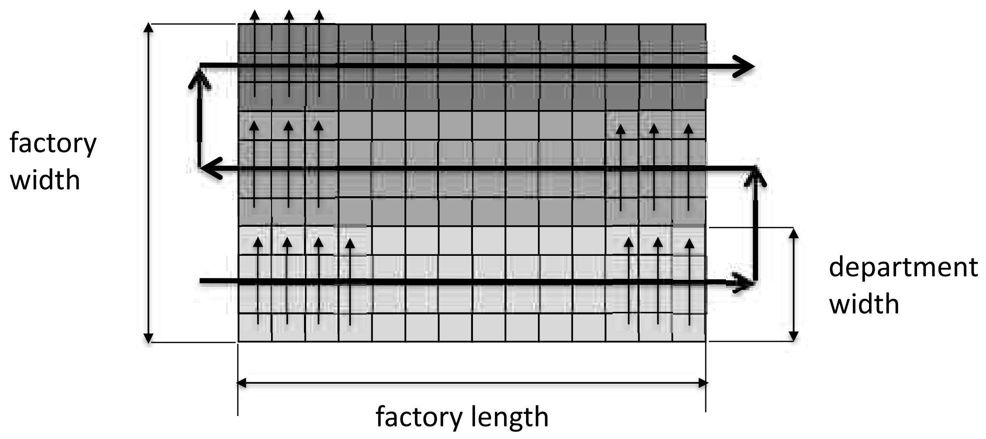


Figure 88 - CRAFT: initial layout generation

Calculation of centroids

Function *Centroids()* calculates centroids of department CX(i) and CY(i), see part of the code in Figure 89. Indexes *i* and *j* represent departments.

```
For i = 0 To L
      For j = 0 To W
            k = p(i, j)
            CX(order_array(k)) = (CX(order_array(k)) + i - 1 / 2)
            CY(order_array(k)) = (CY(order_array(k)) + j - 1 / 2)
      Next
Next
For i = 1 To N
  CX(order_array(i)) = CX(order_array(i)) * cellheight / CN(order_array(i))
  CY(order_array(i)) = CY(order_array(i)) * cellwidth / CN(order_array(i))
Next
```

Figure 89 - CRAFT: Centroids calculation

Adjacency array

In Figure 90, there is a part of the calculation code of the binary Adjacency array *AD(i,j)* where value = 1 means that departments *i* and *j* are adjacent.

```
For i = 0 To L
      For j = 0 To W
            If i < L And p(i, j) <> p(i + 1, j) Then
                  AD(p(i, j), p(i + 1, j)) = 1
                  AD(p(i + 1, j), p(i, j)) = 1
            End If
            If j < W And p(i, j) <> p(i, j + 1) Then
                  AD(p(i, j), p(i, j + 1)) = 1
                  AD(p(i, j + 1), p(i, j)) = 1
            End If
      next
next
```

Figure 90 - CRAFT: Adjacency matrix assigning

Calculation of the distance matrix

Distances *Dist(i, j)* between departments are based on centroids *CX(i)* and *CX(i)*. There are two possibilities of calculation, **Cartesian** and **Euclidian**, as shown in the code in Figure 91.

```
For i = 1 To N
   For j = 1 To N
   If CX(order_array(i)) <> 0 And CX(order_array(j)) <> 0 Then
      If I_ Cartesian =true Then
            Dist(i, j) = Abs(CX(order_array(i)) - CX(order_array(j))) +
            Abs(CY(order_array(i)) - CY(order_array(j)))
      Else
            Dist(i, j) = Sqr((CX(order_array(i)) - CX(order_array(j))) ^ 2
            + (CY(order_array(i)) - CY(order_array(j))) ^ 2)
      End If
   End If
   Next
Next
```

Figure 91 - CRAFT: Distance matrix calculation

b) Optimization procedure

The optimization procedure calculates the total cost of material flows based on distances between centroids (function *Centroids()*, *distanceMatrix()*). If function *switchDep()* returns *savings* greater than 0, then departments are switched by function *Make_Switch*. These procedures are run until no savings are possible (condition savings > 0) or switched departments increase the total cost after re-arranging departments units. The pseudocode of this optimization is in Figure 92, full code is in Appendix (Section A.3.3).

Inputs for the procedure are department information (indexes, sizes as number of units), material flows and costs. Output is an array *p(L, W)* containing department positions.

```
cost=0
savings = 1
lastcost = big
rp = L
cp = W
'********Current layout evalulaion*******:
Calculate: Centroids(), distanceMatrix(), cost = computeCost()
intlcost = cost
Do While savings > 0.1
'************************************
'The last move increased the cost.
      If cost > lastcost Then
            Calculate: Centroids(), distanceMatrix(), cost = computeCost()
            Exit sub 'termination of procedure
      End If
'function switchDep gives department si and sj that is the best for
switching:
      savings = switchDep(rp, cp, si, sj)
      'some savings recalculation:
      If savings > 0 Then
            lastcost = cost
      ' Department switching:
            Make_Switch(si, sj)
      '***Recalculation after switching***:
            Calculate: Centroids(), distanceMatrix(), cost = computeCost()
      Else
            Calculate: Centroids(), distanceMatrix(), cost = computeCost()
      End If

Loop
```

Figure 92 - CRAFT optimization code

Function switchDep()

This function returns department's indexes *si* and *sj* which provide the highest cost saving *SV* from all possible department exchanges. It calculates the total savings of material flows for all possible department exchanges (see Figure 93). A pair of departments is virtually switched and function *switch_savings* calculates cost savings based on the amount of material flows and distances between centroids (Figure 94).

The following input information is used - information about department: $area_{i,j}$, $fixed_{i,j}$ (availability for exchanging, true/false if department is fixed), adjacent array (*AD(i, j)* = if department i and j are in adjacency) and department sequence (*order_array()*).

```
For i = 1 To N - 1
      If $fixed_i$=true And $area_i$ > 0 Then
            For j = i + 1 To N
                  If $fixed_j$, And $area_j$  > 0 Then

                  'Condition equal areas of departments or adjacency:
                        If (($area_i$ = $area_j$) Or (AD(i, j) = 1)) Then

                        S = switch_savings(i, j) 'returns value of savings
                              If S > SV Then
                                    SV = S
                                    si = i
                                    sj = j
                              End If
                        End If
                  End If
            Next j
      End If
Next i
```

Figure 93 - CRAFT code: function switchDep()

Function switch_savings()

This function calculates the total cost savings (*switch_savings*) of all material flows if departments *si* and *sj* are switched (see Figure 94). Material flows between departments are in array *InterCost (i,j)*.

```
switch_savings = 0
For k = 1 To N
  If (k <> si) And (k <> sj) Then
  FI = InterCost(k, si) + InterCost(si, k)
  FJ = InterCost(k, sj) + InterCost(sj, k)
  switch_savings = switch_savings + (FI - FJ) * (Dist(si, k) - Dist(sj, k))
  End If
Next k
```

Figure 94 - CRAFT code: function switch_savings ()

Function MakeSwitch()

Function *MakeSwitch* switches department units. The full code is in Appendix (Section A.3.3).

c) Placing department unit block into layout

The last phase of CRAFT methods is to insert the drawing blocks that represent department to the layout. Input data is a 2-D array *p(L,W)* (L is length, W is width of layout) containing ID numbers of departments. The layout generation is based on the procedures *"Inserting blocks and assigning attributes"* described in Section 4.5.2. Departments blocks are distinguished by attributes "DEPARTMENT_NAME", "ALT.:" (alternative name), colour and layer ("CRAFT").

4.5.5 CORELAP

The CORELAP placement method is based on the relationship between departments. Heuristics of this method is described in Section 2.2.

The development of CORELAP consists of several phases:

- Loading and preparing data
- Input sequence preparation
- Layout building
- Placement of blocks into the layout

a) Loading and preparing data

Relationship data are loaded from database for the chosen alternative. Array *rel_table_AEIOX()* contains an alphabet expression of relationships. These values are recalculated into number values based on the alternative (e.g. "A"=64) and assigned to array *rel_table()* . Relationships between the same departments have a value that equals 0.1. It helps to make the final department shape more rectangular and regular.

b) Input sequence preparation

In this phase, the input sequence "*sequence()*" of department is established, based on the importance of the relationship. In Figure 95, there is a short-cut code for it. Firstly, the department with the highest TCR value (*Total closeness value*) is selected and it is assigned in the first position of the *sequence*. Then, department with the highest TCR value with already placed departments (involved in the *sequence*) is selected and assigned to the *sequence*. This process is repeated until all departments are assigned in the *sequence*.

If there is an X relationship, the procedure jumps (*x_jump*) that department and is assigned to the last position of the sequence. If there is a tie (the same TCR values), TCR values are recalculated and the department with total highest TCR is selected.

```
'procedure: Searching department with highest total sum of relationship
sequence(1) = akt_max_index
entered(akt_max_index) = True
number_entered=0

'***repeat until all departments are entered***
Do Until number_entered = N
booFound = False
'check if department was already entered, as department with x-
relationship:
For i = 1 To N
      If entered (i) = False Then
'searching for relationships and TCR values, already placed department in
the sequence(k):
            For k = 1 To N - 1
                  boo_jump = False
'X relationship - value < 0:
                  If rel_table(i, sequence(k)) < 0 Then Then
                  akt_max_index = i
                  boo_jump = True
                  GoTo x_jump
                   Else
                  'array akt_value(i) involves TCR
                  akt_value(i) = akt_value(i) + Abs(rel_table(i,
                  sequence(k)))
                  booFound = True
                  End If
             Next k
      End If
Next i

'find of maximum in the array akt_value(i):
            If booFound = True Then
                  akt_max_index = 0
                  For i = 1 To N
                        If akt_value(akt_max_index) < akt_value(i) Then
```

```
                              akt_max_index = i
                              akt_max_number = 0
                              ReDim ties(N) ' array ties() contains ties
                        End If
                        If akt_value(akt_max_index) = akt_value(i) Then
                              akt_max_number = akt_max_number + 1
                              ties(i) = i
                        End If
                  Next i

'***Ties***:
            If akt_max_number > 1 Then

            ReDim akt_value(N)

'find the rest of realationships:
                 For k = 1 To UBound(ties)
'TCR values for department with ties:
                  If ties(k) <> 0 Then
                   For i = 1 To N
                         akt_value(k) = akt_value(k) + Abs(rel_table(k, i))
                   Next i
                  End If
                 Next k

'procedure: Find maximum (akt_max_index=i) in the array akt_value(i)
            End If 'tie

            x_jump:
            ReDim akt_value(N)

            If boo_jump = True Then
'negative relationships x
                  entered (akt_max_index) = True
                  number_entered= number_entered+1
                  x_counter = x_counter + 1
'department are placed in the last position of sequence:
sequence(UBound(sequence) - x_counter) = akt_max_index


                  j = j - 1
                  boo_jump = False
            Else
'positive realationships:
                  entered (akt_max_index) = True
                  number_entered= number_entered+1
                  sequence(j + 1) = akt_max_index
            End If
            boo_jump = False

            End If 'booFound

Loop
```

Figure 95 - CORELAP: Input sequence preparation

c) Layout building

In this phase, the layout is generated based on the sequence of departments and their sizes (a number of units). As well as in the CRAFT method, the layout is represented by the 2D integer array (*Plant_dep(x,y)*) where array elements mean departments. The procedure firstly allocates segment into the array based on the sequence so that units of the same department are being assigned step by step. After achieving the appropriate number of units, the insertion continues by assigning the next department units. The layout, consisting of drawing blocks, is generated based on this array. In this section, only the heuristics is described (see Figure 96). A full code is in Appendix (Section A.3.3).

```
Sequence();Department_size(); Department_size(x,y)
k=1
Repeat
      department =Sequence(k)
      size=0
      Repeat
***Select optimal position in the grid (array) for given department***
            Call Function_Evaluation_All_Space(department)
            Call Function_Find_Maximum(return values: max_x_array,
            max_y_array)
***Place department index into plant array Plant_dep((x, y)***
            Plant_dep(max_x_array, max_y_array) = department

            size= size+1
      Until all units of department are placed (size= Department_size(k))
            k=k+1
Until all department from sequence are placed (k=N)
```

Figure 96 - CORELAP: Layout building heuristics

Function_Evaluation_All_Space(department)

All non-allocated segments in the already assigned departments of the *Plant_dep(x, y)* are searched and their surroundings segments are evaluated by TCR values based on the relationship chart. Surroundings are 8 segments around the non-allocated segment (higher left, higher, higher right, etc.). Example of the code is in Figure 97.

The evaluation function prefers segments with not already achieved relationship. This information is in the binary array *AD_dummy(department, department)* based on the relationship chart. Value ”0” means that there is no relationship or the relationship has already been achieved; value 1” means that a relationship has not been achieved. This array is upgraded after each allocation by function *“AD_dummy_upg()”*. *Coeficinet_AD* forces the establishment of a new relationship, if the already achieved relationships are strong and total values in the *TCR_values()* are higher than the new relationship TCR value. *Alfa_adj_coef* is a coefficient that considers department neighbouring by edge (equals 1) or by corner (equals 0.5).

```
For xi = 1 To UBound(Plant_dep, 1) -1
      For yi = 1 To UBound(Plant_dep, 2) -1
            'not allocated segments:
            If Plant_dep(xi, yi) = 0 Then
            'higher, left segment:
                  If Plant_dep((xi - 1), yi + 1) >= 0 Then
                  TCR_values(xi, yi) = TCR_values(xi, yi) + alfa_adj_coef *
                  rel_table(Plant_dep((xi - 1), yi + 1), depart_fce)
                  +AD_dummy(Plant_dep((xi - 1), yi + 1), depart_fce) *
                  coeficinet_AD * rel_table(Plant_dep((xi - 1), yi + 1),
                  depart_fce)
                  End If
            'higher, right segment...
            End If
      …
      next
next
```

Figure 97 - CORELAP: Function_Evaluation_All_Space()

Function_Find_Maximum()

This function returns coordinates x and y where the maximum *TCR_values(x,y)* is.

d) <u>Placement of blocks into layout</u>

This procedure is the same as in the CRAFT method, described in Section 4.5.3 - Placing department unit block into layout. Each department has a specific colour. Helpful graphic objects are also generated. A relationship chart is inserted with highlighted achieved relationships. Relationships can be generated and displayed as coloured distinguished flows connecting centroids of departments. The total score of the achieved relationships is calculated for an easy comparison of alternatives.

4.5.6 Graph Theory Block Layouts

The Graph Theory method optimises resources placement based on material flows. In IDS, it is implemented a modification of this method in which resources are placed into a triangular grid. It is described in Section 2.2.1, as well as its heuristics. An example is presented in Section A.4.6. The implementation follows and has the following phases:

- Preparing input data
- Sequence of resource
- Placing resource into triangular grid
- Inserting drawing block into layout

a) Preparing input data

Input data set is a *"from- to"* chart or a *"between-to"* table where flows are summarised between a pair of resources *from* and *to*.

b) Sequence of resources

Based on the material flows, a *Sequence* of resource is established. Firstly, the maximum material flow is searched and appropriate resource pair is assigned to the first two positions of the *Sequence*. Then, the resource, which has the largest sum of flows with the already selected resources, is chosen and saved to the *Sequence*. Full code of function *"Triagular_rule_department()"* is included in Appendix (Section A.3.3).

```
Sequence(N); material flows from_to(N,N); Plant(2*N , 2*N)

k=1;
x= 2*N/2; y= 2*N/2
Plant(x, y)= Sequence(k)
position_x(k)=x; position_y(k)=y

k=2
position_x(k) = position_x(1)-1; position_y(k) = position_y(1)
Plant(position_x(k), position_y(k))= Sequence(k)

Repeat
'evaluation of each segment in the  Plant(x,y)
for x =1 to 2*N
      for y =1 to 2*N
'not allocated segment:
            if Plant(x,y)=0 then
            i=k
'each segment is evaluated by a material flow*distance
            Repeat
                  Plant_value(x,y) = Plant_value(x,y) +
                  from_to(Sequence(k),Sequence(i))*
                  Distance([x,y]; [position_x(i), position_y(i)]
                  i=i-1
            until i=0
            end if

      next y
next x

Procedure: Find minimum value in the array Plant_value(x,y), return
coordiantes xmin and ymin

Plant(x_min,y_min)=Sequence(k)
reset Plant_value()
k=k+1
Until all department are placed (k=N)
```

Figure 98 - Graph Theory Layouts: placing resource to the triangular grid

c) <u>Placing resource into triangular grid</u>

Resources are assigned step by step from the *"Sequence"* to the 2D array *"Plant()"* representing layout (Figure 98). The first resource from the *Sequence(k=1)* is assigned to the middle of the array, then the second resource is assigned to the left position of the first. After that, the procedure continues until all the remaining resources are assigned. Each segment of the *Plant(x,y)* is evaluated by a value saved in the *Plant_value(x,y)* - material flows between the currently assigning resource *k* and the already assigned resources, times distance between segment *Plant(x,y)* and positions of the already placed resources *position_x(i), position_y(i)*. After that, the minimum value of *Plant_value(x,y)* is founded and the resource is assigned there.

d) <u>Inserting a drawing block into the layout</u>

A layout drawing is generated based on the array as described in Section 4.5.2. Coordinates must be recalculated from triangular to Cartesian coordinate system for the purpose of AutoCAD. There are also generated material flows by function *"Flows_triangular_rule()"*.

4.6 User control forms

A great effort has been made in order to make IDS more accessible to the user. Therefore functions, tables and methods mentioned in the previous chapters are implemented into user-friendly forms. These forms are helpful for input /output data, controlling of functions, data analysis, checking obvious logic fails, errors and input data, displaying data in graphs, automatic filling of some data-fields, etc. Forms are included in MS Access and in AutoCAD; WITNESS contains functions under the menu **USERACTION** and **INITIALISATION**. The full list of forms and their structure can be found in the Table 22. This table includes numbers of forms that are shown in Figure 100 and Figure 101. From those forms, the complete IDS can be controlled.

Description of forms usage and functions is explained in Appendix 2. That chapter contains a set of lectures and tutorial of IDS system.

General issues about system objects, their properties and functions, such as resources, storages, entities, crates and operations (see Section A.4.2) are described. The use of simulation functionalities, such as generation of simulation models, initialisation setup, placement of objects in simulation model or stochastic functions, is described in Section A.4.3. The use of material flows is explained in Section A.4.4. As mentioned before, IDS supports automatic generation of layouts. That can be factory layouts (Section A.4.5) based on CRAFT, CORELAP or Graph Theory Block layouts. Layouts of facilities can be based on simplified approach without considering detailed operation data and storing data (see Section A.4.8) or layout based on simulation data. Basic layout patterns can be generated - cellular, process layouts and production lines (see Section A.4.9) as shown through an example. Two different approaches of transport and supplying systems are explained with examples in Section A.4.10 (forklift) and Section A.4.11 (milk-run). A full production system design example using IDS is illustrated in Section A.4.12, including supplying, unpacking and assembling operations.

The mentioned principles and functionalities are used in the real sized projects described in Chapter 5.

Table 22 – IDS forms structure

Tool	Main form name	Forms				
DB (1)	CAD_Library (2)	CAD_Parts	CAD_Crates	CAD_Inventory	CAD_ HR	CAD_Space
	Resources (3)	Parts	Crates	Inventory	HR	Space
	Operation_setup (4)	List_of_Operation	Input_Output	Process_order		
	Work_possibilities (5)	Inventory-Operation	Operation-Inventory	Inventory-Human_Resource		
	Place_setup (6)	Table_Storage_Machine	Table_Machine_Storage			
	Transportation (7)	Transporters	Milk_run	Time_path	Conveyor	
		Transport_operation	From_to_table			
	Stochastic (8)	List_of_profiles	Profile_Detail			
	Part_Scheduling(9)	Part_Arrivals	Table_part_schedulling			
	Weight_Parts (10)	Table_Weight				
	Generation_SIM models(11)					
	Simulation_results (12)	Processes_results	Space_results			
		Material_flow_results	Part_throughtput			
	Material_flow_analyse (13)					
	Production analyse(14)	Product_Quantity_analysis	Cluster_analysis			
	Administration					
	Factory_layout_design(15)	Table_department	Alternative_properties			
	Deterministic_models (16)	Departments	Parts	Schedules/Input amount		
CAD	Factory	Alternative_settings (17)	Batch (22)	Setup (23)		
		Manual (18)				
		CRAFT (19)	CORELAP (20)	Graph method (21)		
	Facility	General (24)				
		CAD library(25)	Blocks (26)			
		Material Flows (Generation of Material flows, Displaying of material flows) (27)				
		Facility Placement (28)				
		Buffer solution (29)	WITNESS (30)	Transport function (31)	Setup (32)	
		Simple models (33)				
WITNESS	Menu: User Action	fce_Finish_statistics () fce_Write_Storage_Data () fce_Position_objects ()				

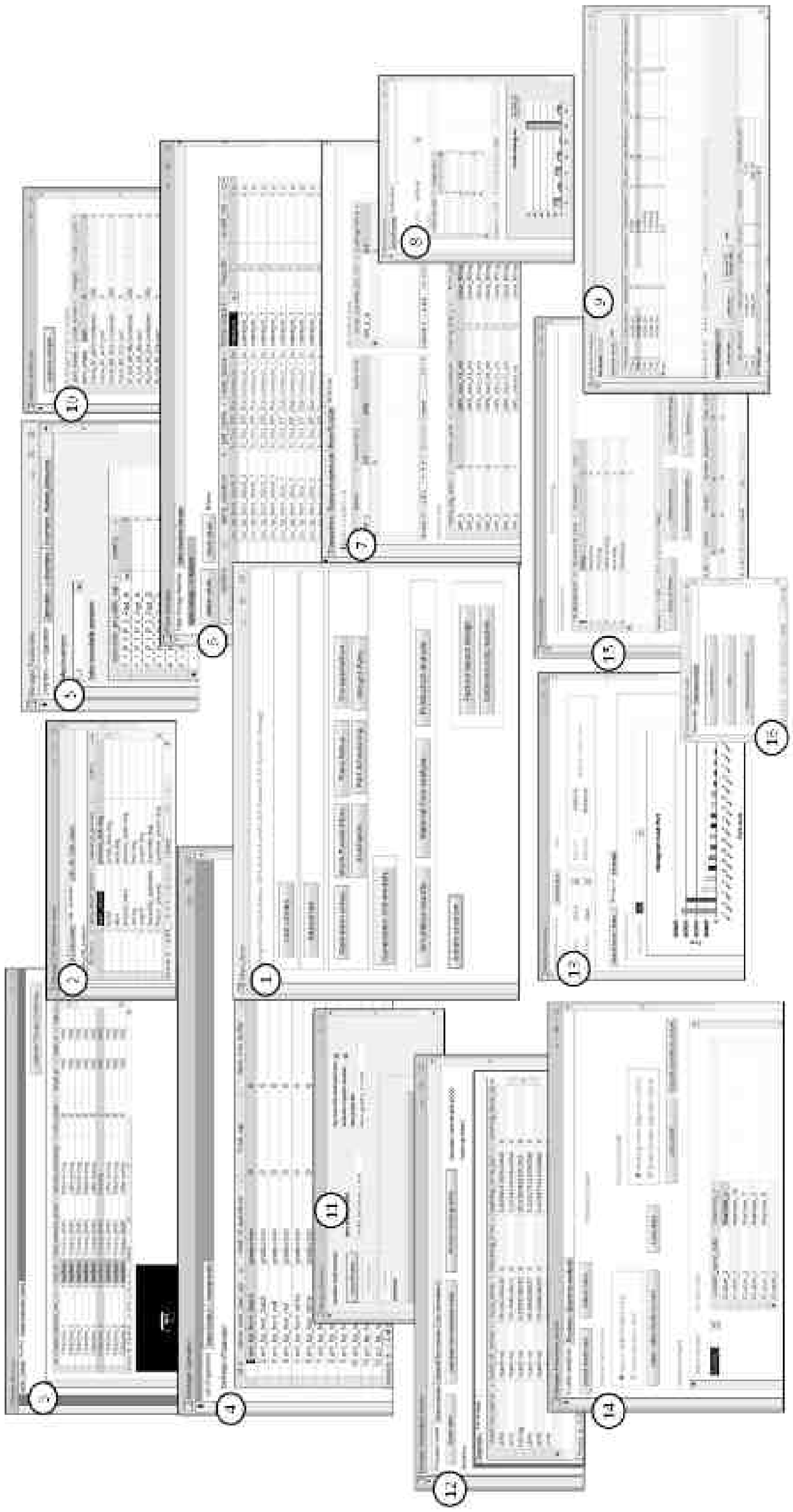


Figure 99 - Input/Output forms MS Access

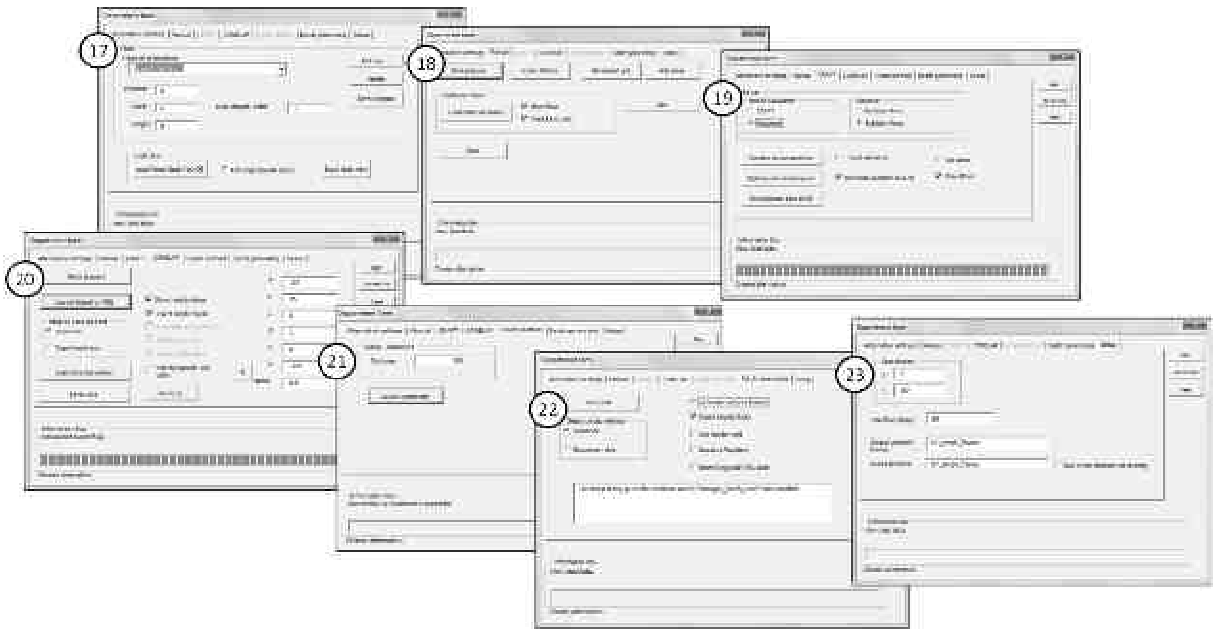


Figure 100 - Input/Output forms AutoCAD: Factory

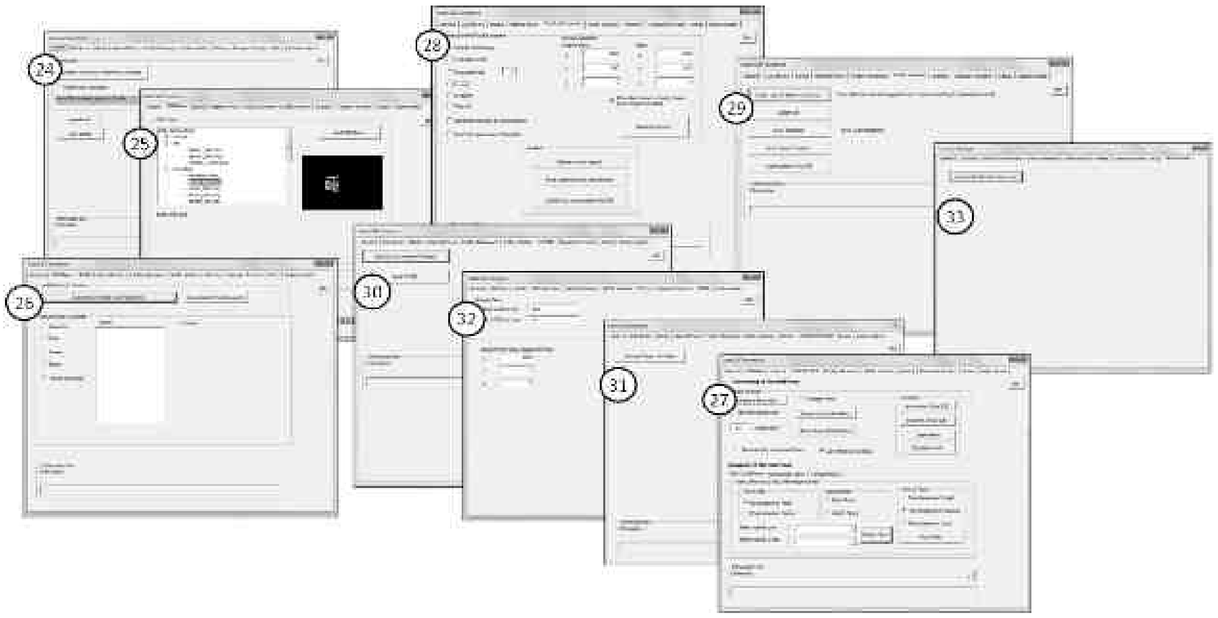


Figure 101 - Input/Output forms AutoCAD: Facility

4.7 Transport systems design

The design of a transportation system in the factory or in the facility is a common task. Transportation means moving entities from loading to the target position. There can be two basic kinds of transportation: taxi and train based transportation (Tanchoco, 1994), (Taylor, 2009), (Halevi, 2001).

Taxi principle says that the transporter is called for a specific operation. It can be a **forklift**, a crane, etc.

Train principle says that the transporter visits all the stations in the defined cycles; the usual representation is a **milk-run** (MR) (see Figure 102).

MR supplies stations (customers) with items loaded in the warehouse. It has strictly defined routes and delivering schedules (Satoh, 2008)(Jiang, Huang, & Wang, 2010). It can be compared with the subway train that is directed by a defined schedule: it stops at each station, takes in a few passengers while a few get out. This principle can be used in internal and external logistics. The most used facilities for MR are the logistic trains that consist of one engine (locomotive) and a few wagons (www.fantozzi.com, 2011).

The idea comes from the past when cars from milk farms would deliver milk at an exact time. It is very useful when associated with the kanban system, empty units from stations are picked up and directly changed by loaded units (Krieg, 2005).

Empty transportation boxes are changed by loaded boxes in the warehouse. A unit can be represented by a transportation box and a kanban card.

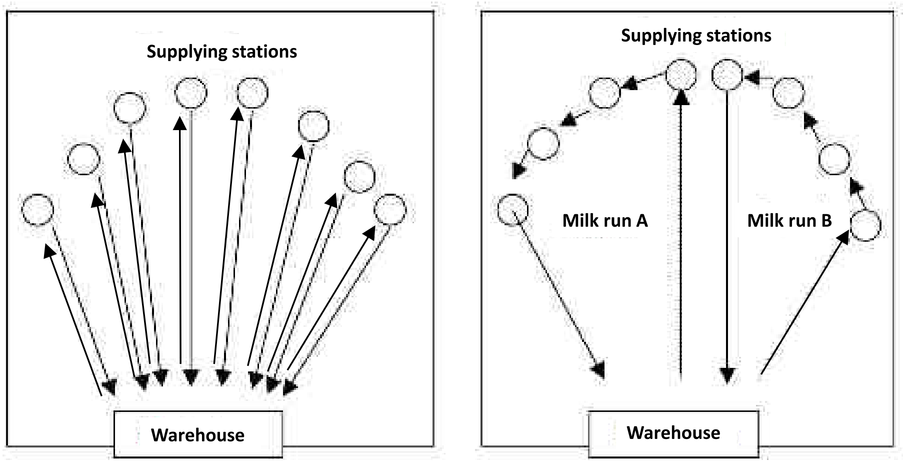


Figure 102 - Transportation: Taxi and train approaches (source Krieg, 2005).

IDS includes a set of powerful functions for designing transportation systems in an integrated approach. Based on the data from the database, a simulation model is generated. This model contains specific transportation functions. In the CAD layout, an optimal transportation network can be designed. An optimal transportation system involves:

- The kind of transportation and control politics (e.g. manual handling, conveyor, forklift, milk-run, etc.)
- The number of transporters
- Transportation routes
- Space limitations and manoeuvrability
- Transportation unit and capacity
- Stochastic influences

The general approach for the design of transportation systems is shown in Figure 103. Material flows are generated in the layout (1) according to the transportation network. The transportation network is designed based on the roads and aisles between facilities. The position and length of these flows (2) are recorded to DB (3). Based on these data and transporters speed, transportation times are calculated (4). The generated simulation model (5) verifies it and the results of the simulation experiments (use of forklift, throughput, etc.) (6) are recorded in DB (3) for analysis. If it is required, a new iteration cycle can be started with updated values (1).

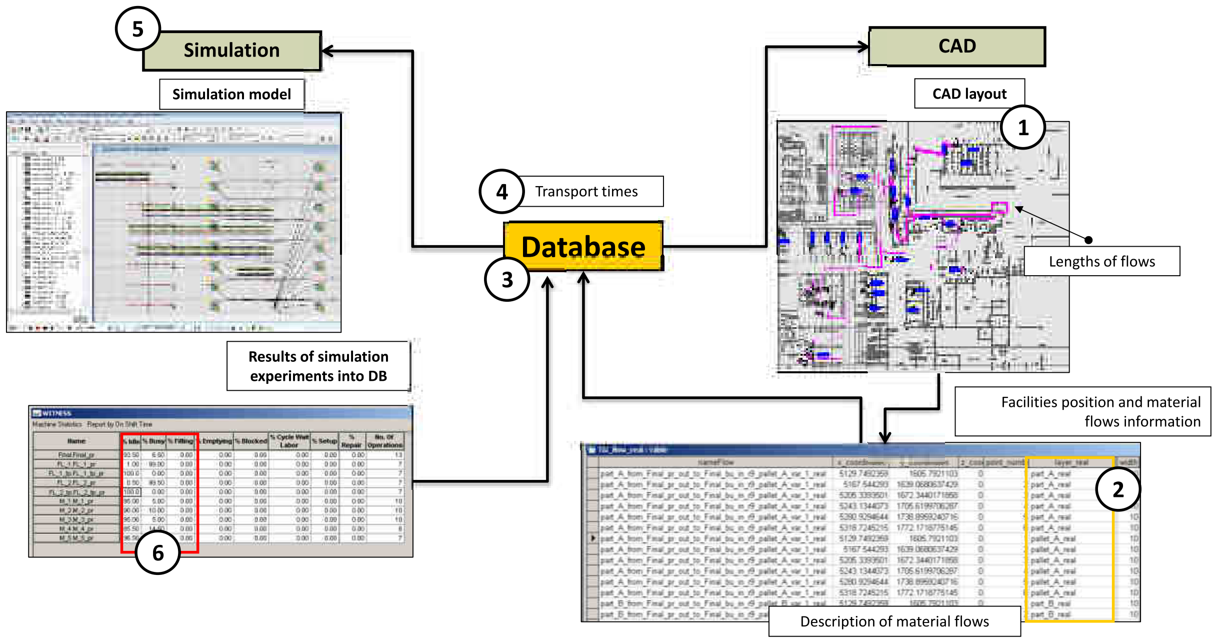

5
Simulation
Simulation model
CAD
CAD layout
1
4
Transport times
Database
3
Lengths of flows
Results of simulation experiments into DB
Facilities position and material flows information
2
6
Description of material flows

Figure 103 - Design of a transportation system

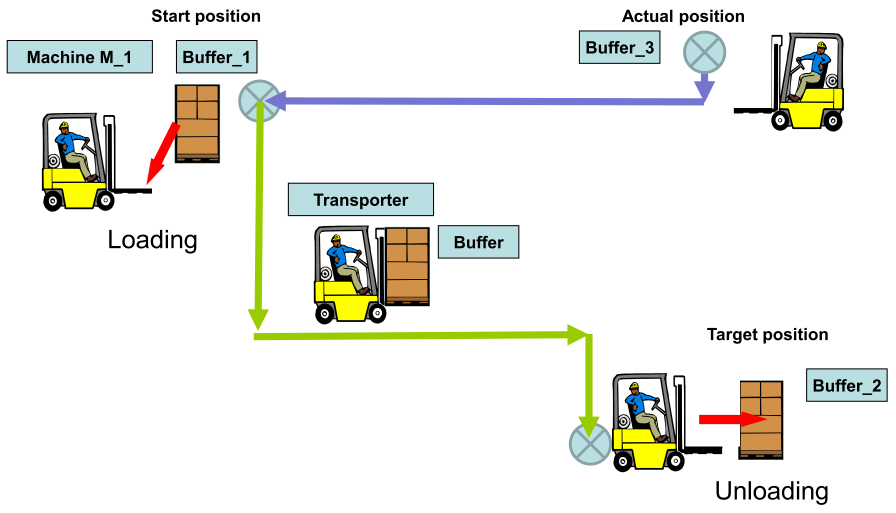


Figure 104 - Principle of taxi transportation

4.7.1 Design of a taxi transportation system

A basic schema of a transportation system is in Figure 104. Transportation operation consists of moving products (in this example, pallet with boxes) from “Buffer_1” to “Buffer_2” with a transporter (forklift). The transporter can only move on the transportation network of aisles and its position can vary.

The complete transportation operation involves two sections:

- movement from the actual position to the start position (loading)
- movement from the start position to the final position

The description in the simulation model uses the following concept: a resource cannot make any movement so the transporter is substituted by a "transportation helper entity". Transporter resource consists of two units: one for controlling arrival instants and one for transportation of loaded products.

The transportation operation is composed of the following sections:

- time of transportation helper from the actual position to the start position (blue arrows)
- loading time (assembly of transportation helper and loaded products) (red arrows)
- transportation between start and target position (loaded products green arrows)
- unloading (disassembly of transportation helper and loaded products)(red arrows)

The beginning of the transportation operation is based on the arrival of products that will be transported to the start position.

In the IDS simulation models, additional functions and variables are used: *"Fce_transport_time_tp()", "Fce_transport_time()"* and *current position* (location) of transporter [Resource][_ap], as described in Section 4.4.9.

This functionality can be joined to the layout data using distances from the layout drawing. These values can be taken as a direct distance between two resources or as material flows based on the transportation network. These flows consider aisles, shape constrains, etc. that correspond to real conditions.

4.7.2 Design of a train transportation system

Another kind of transportation is a train system where the transporter visits all defined stations during its cycle. The usual representation is a milk-run system (MR) that supplies the production lines. This type of supplying has been increasingly wide spreading because of its effectiveness, mainly with a kanban pull system. The design of MR has to consider several factors as distances of stations, routes of MR, delivered product volumes, product variety, cycle time, etc.

All these information can be specified by IDS and its integrated tools.

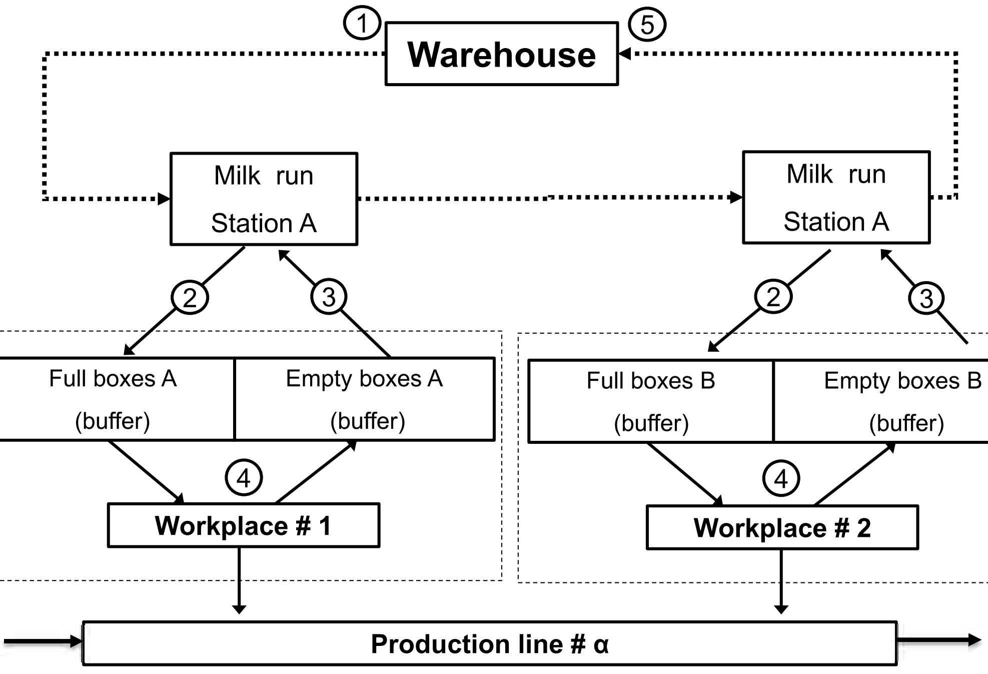


Figure 105 - Principle of MR

There is a production line # α which consists of several workstations and that are supplied by products transported in boxes. Items are stored in the Warehouse. The purpose of MR is to deliver products to the given stations (dotted line). Full boxes with products are loaded (1) and delivered to the appropriate buffers in the stations (2). Empty boxes are taken back (3). Parts are taken from boxes and processed in the Workstations (4). After passing all stations, the empty boxes are changed by full (5) and a new cycle can start.

The simulation model consists of several functions for describing the MR behaviour: *„function_Milk_run()"* makes definition of MR resources; *"function_MR_load_cy()"* holds the number of passed cycles; *„function_Change_io()"* controls the exchanging of amount of empty boxes by full boxes. Codes of these functions are in Appendix (Section A.3.3.3).

Input/output number of entities in production operation is a constant value during the simulation running. In the case of MR´s operation, the input/output number of entities is changing. This cycle of transportation units can be defined as the following sequence:

- load full boxes in the warehouse
- unload full boxes in the station
- load empty boxes in the station
- unload empty boxes in the warehouse

The MR system can be defined as a transportation cycle where all stations are visited. This system consists of sub-cycles where the amount of boxes is filled.

The cycle time of MR is usually a constant value and consists of loading boxes, delivering and changing empty by full boxes. In Figure 106, there is a schema of MR states. The busy time of MR can differ, so the rest of the MR cycle period (100 units) is spent waiting in the Warehouse. This time is calculated as:

```
Waiting time = MR_cycle * #MR_cycle - Time_actual,
```

where MR_cycle is a MR cycle time, #MR_cycle is the number of performed cycles.

In the simulation model, waiting time is implied by a service time of MR in order to make it easier to distinguish different state of MR and to have an easy definition.

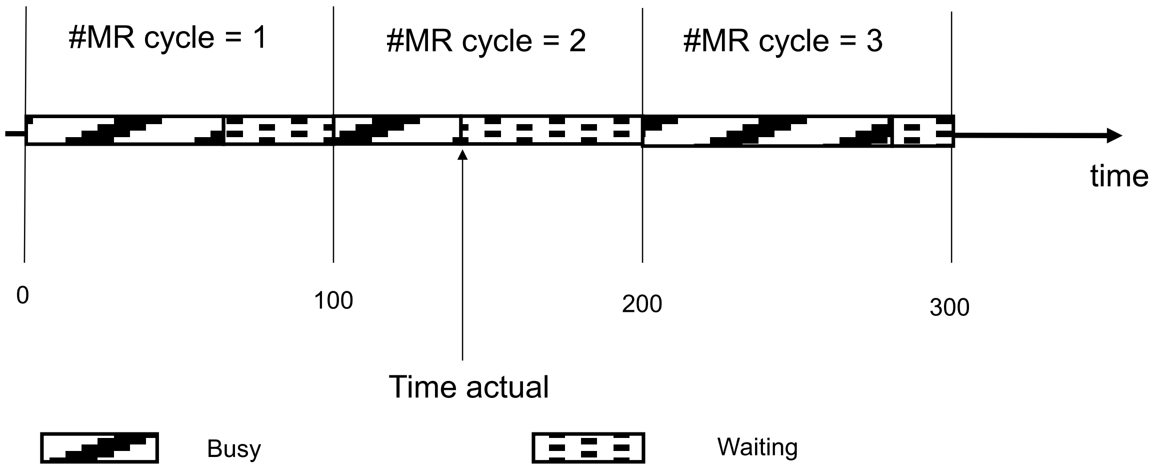


Figure 106 - MR cycles

By this solution, it is possible to implement breakdowns of MRs, to compare different alternatives, to establish the cycle times, the number of MRs and their capacity (wagons), to analyse the MR statistics states, etc. The integration of tools provides the existence of routes distances from the CAD layout and the design of optimal routes.

4.7.3 Transportation network design

The transportation system consists of designing the transportation network and finding the shortest path for two stations (start and final). In order to find the shortest path in the transportation system the *Dijkstra* algorithm can be used. This was chosen due to general application usage and easy implementation in IDS and in AutoCAD (Cormen, 2001) (Sniedovich, 2006).

Input sets of data are start and final stations and the transportation network designed in the layout drawing in the specific layer *"transport net"*. It involves all the transportation routes, aisles, paths, etc. and which way the transporter can go.

The practical use of this algorithm is shown and described in Section A.4.5, e.g. in Figure 215.

The full programme code of this well-known algorithm is in Appendix (Section A.3.3).

a) <u>Implementation of the shortest path algorithm into the Displaying of material flows:</u>

The following code is implemented into the generation of material flows (Figure 107). Function *"Dijkstra_Points()"* is called with inputting coordinates x, y, z of starting point (*"From_point"*) and final point (*"To_point"*) representing start and final stations. These coordinates of points can be rounded due to the used precision in AutoCAD. As output of this function, array with coordinates is returned (*"arrCoordinates_temp"*). Based on it, polyline is drawn in AutoCAD.

```
precision_round = 1

If boo_Dijkstra = True Then

      Call Dijkstra_Points(Round(From_point_x, precision_round),
      Round(From_point_y, precision_round), Round(From_point_z,
      precision_round), Round(To_point_x, precision_round),
      Round(To_point_y, precision_round), Round(To_point_z,
      precision_round), arrCoordinates_temp)

      arrCoordinates = arrCoordinates_temp

      Set objEnt(0) = ThisDrawing.ModelSpace.AddPolyline(arrCoordinates)
End If
```

Figure 107 – Dijkstra algorithm implementation

4.8 Iterative buffer sizing

System IDS supports automatic design of storages or buffers based on data from simulation experiments combined with CAD layout limitations. This approach is shown in Figure 108. From the simulation experiments (1), a maximum number of parts in the given buffer can be taken as initial buffer size (**IDS.DB.RESOURCES.*<u>Space.capacity</u>***) Also, those values can be established by the analysis of historical data (described in Section A.4.2.). Values are recorded to DB (2) into fields *Maximum_Total_Sim*. These values can be copied to the fields *Actual value* (3) for the use in the CAD layout. Those numbers of parts can be inserted as the CAD blocks into a specific buffer. This process is described in Appendix (Section A.4.5). After arranging these blocks (5), the buffer limitation is established by CAD and recorded to DB (6). These values can be copied again to the fields *Actual value* for validation by a new simulation experiment.

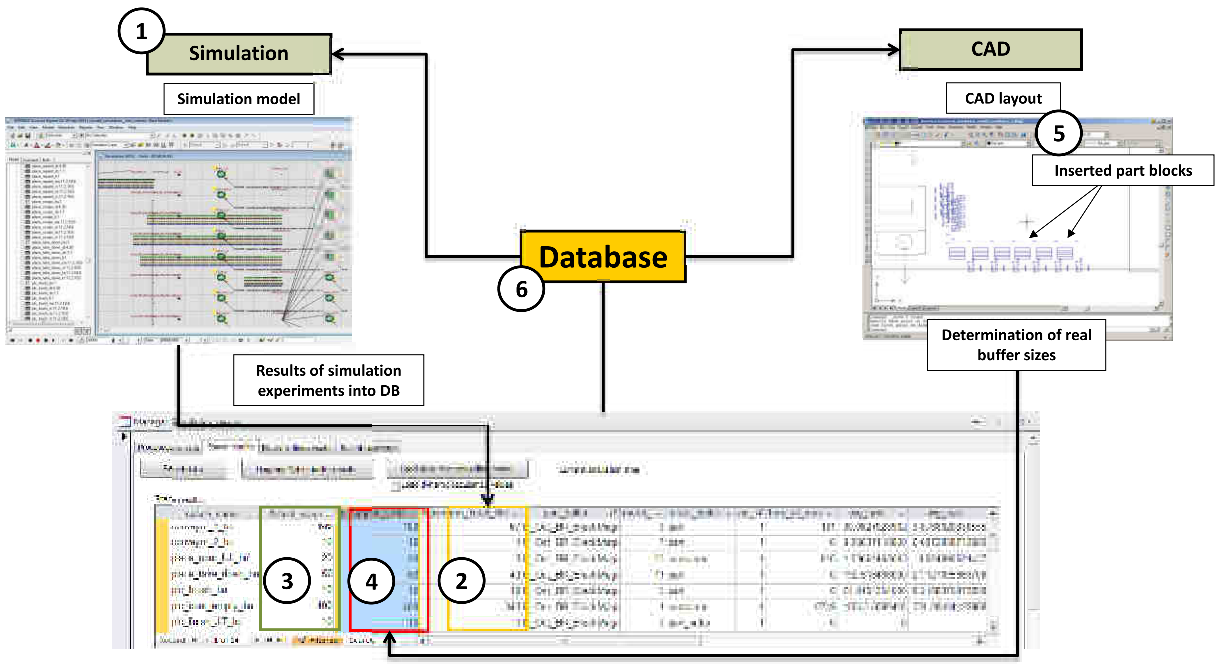

1
Simulation
Simulation model
CAD
CAD layout
5
Inserted part blocks
Database
6
Determination of real buffer sizes
Results of simulation experiments into DB
3
4
2

Figure 108 - Iterative storage design

4.9 Summary

In this chapter, the development of the IDS concept is presented - selection of software tools and methods, the integration approach, the structure of the database in MS Access, the principle of automatic generation of a simulation model in WITNESS and its features. The work on basic CAD elements and their application to layout design, on material flows displaying and on layout optimization methods is described. In the next chapter, the implementation of mentioned functionalities on practical examples is presented.

5 IDS implementation

The following examples demonstrate the practical use of IDS on real sized projects from different and various areas of industry (see Table 23). The main purpose of these projects is not to present specific results of each project but illustrate the approach of systems design and analysis using IDS, compared with traditional ways, e.g. manually built simulation models. This is also main reason for choose proposed examples, exploring different functionalities of IDS in each one of them. Data and IDS functions are controlled by user forms described in Appendix 4.

Table 23 – Projects overview

Company/Project name	Area	Focus/topic
Cachapuz Company	cement plant	material flows control promotion
Blaupunkt Bosch	automotive industry	milk-run system kanban system
Magna (Cadence Innovation)	automotive industry	batch production and JIT assembly
Modus	producer of lights	job shop system, re-layout
University Campus Design	theoretical example	CORELAP and CRAFT arrangement of departments Simulation model of restaurant Level in designing (from general to detailed models)

5.1 Cachapuz Company

5.1.1 Company overview

Cachapuz Company is a member of The Bilanciai Group, one of the biggest European groups on **weighing systems** (www.cachapuz.com, 2011).

It has a strong market position in the automation of logistic operations in the cement, agriculture and other kind of factories. Cachapuz is a producer of hardware logistic tools as weight bridges (WB, systems measuring weight of trucks), traffic control, gate systems, etc. It has also developed a software logistic system (*SLV)* for the control of these tools and self-service driver mode. During more than thirty years of history, the Cachapuz solution has been spread all over the world (EU, Angola, Egypt, Brazil, etc.).

5.1.2 Project introduction

This project describes the logistic system implementation in a **cement plant** in Jaypee – Rewa, India (www.jalindia.com, 2011). Because of lack of some data, another cement plant, similar in size, structure and technical equipment, was used to collect important data (Alexandria, Egypt).

This project involves the following phases:

1. **Design of optimal system configuration**
2. Installation of hardware facilities and logistic tools
3. Implementation of the complete logistic system (hardware components and software SLV Cement)

The first phase of the project was solved in 2009 by the simulation tool SIMIO and also by IDS (Vik et al., 2010a)(Vik et al., 2010b)(Vik, Dias, Pereira, & Oliveira, 2010c).

5.1.3 Description of the system

a) SLV Internal Logistic System

The SLV Cement is a complete logistic system for cement plants developed by Cachapuz. This automatic and integrated logistic system deals with all the processes, since the arrival of a truck to the cement plant and to the shipping of the cement. At the end of the process, even the administrative tasks, as issuing and printing the necessary documentation, are performed automatically. Another interesting SLV feature is the complete integration with SAP business software (ERP)(www.slvcement.com, 2011).

SLV architecture is based on the central core and set of modules, such as reporting, automation and control, alerts (with SMS and email sending), process logs and extensibility with external ERPs. Over these modules, the different plant areas like parking, check-in / check-out, raw material unloading zones, bag warehouses or bulk-loading, will be customized and automated.

These modules are fully integrated logistic system that will permit to increase throughput (number of trucks per defined time) and reduce the number of operators needed, thus avoiding human mistakes.

The SLV Cement uses a set of hardware devices of the logistic control – as electric information boards, registration kiosks (touch screens, printers, and RFID card readers), access cards for customers (truck drivers), traffic lights, positioning sensors and counters, etc.

With this configuration, it is possible to achieve self-service operations for 24 hours, 7 days a week, and more than 100 vehicles dispatched in one hour, as in the cement plant in Alexandria, Egypt.

b) Cement plant description

These plants produce and sell a basic building material, which is cement. Traffic and material flows in these types of plants can be enormous. More than 500 trucks per day, each with more than 50 tons of material, supply a medium sized cement plant. More than 250 trucks with 30 tons of cement are loaded every day based on data from the SLV system, in a similar plant size in Alexandria, Egypt. The need for an efficient logistic system is crucial.

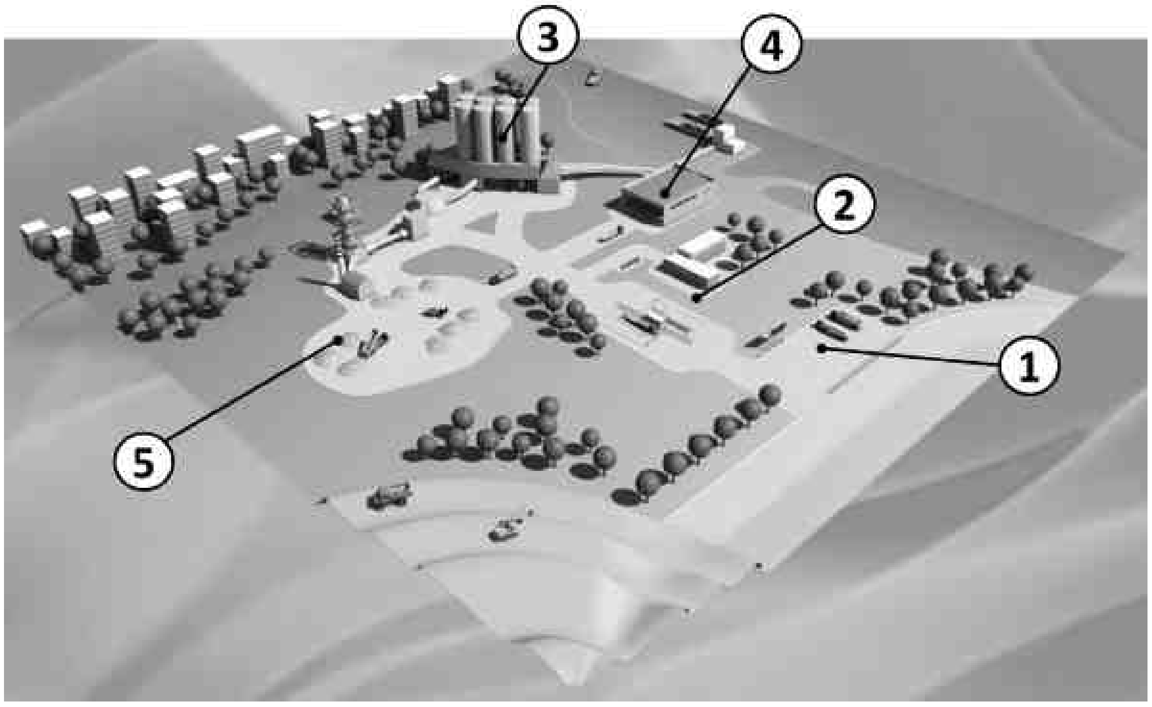


Figure 109 - Configuration of a cement plant (source: www.slvcement.com, 2011)

Figure 109 shows a typical **configuration** of a cement plant with the identification of the basic areas. The **parking area** (1) is an external zone where trucks wait to be called, after the registration. **The entrance and exit gates** (2) are check in/out zones, usually with **weight bridges** (WB). WB is the equipment that measures the weight of the trucks. Weight is measured when the trucks arrive and when they leave the plant. Examples of some equipment are in Figure 123.

The silos (3) are equipment for loading the cement product into trucks (bulk). Another way of loading cement is through bags in the **warehouse** (4). This type of loading can have a low automation level (conveyor and human power) or a higher level with the use of pallets, depending on the type of commerce and market requirements. The cement plant is supplied with raw material that is stored in the **raw material areas** (5). It is usually convenient to separate these processes because the respective large number of trucks can cause serious flow problems inside the plant.

c) <u>Problematic areas and general targets in the logistics</u>

Based on a high number of case studies done by Cachapuz, it is possible to identify general problematic areas and processes and its respective project aims. **Loading processes** are the most critical in the cement plant, mainly the process of loading cement bags in the warehouse. As far as the correct type of loaded material and correct amount of material is concerned, this area is usually responsible for most process failures. Due to the long operation time (in case of manual loading), a bottleneck of system appears. It leads to a **long waiting time** of customers in the queues. The inadequate traffic control of trucks leads to traffic jam inside the plant (overcrowding) and to misunderstandings regarding the destinations of trucks. In the other hand, there can also be trucks waiting outside factory, even when facilities are available. Other issues are the absence of new and modern technology, as registration done by electronic cards (administration), or the breakdowns of old facilities.

5.1.4 Project aim

As mentioned above, this project is focused on finding the optimal system configuration of the cement plant and it demonstrates the advantages of using the SLV components to improve the current system. For this purpose, IDS is used and the following tasks were established:

- Analysis of the current system consisting of material flows, schedules, etc.
- Finding a high performance configuration and control of the **logistic components** in cement plants
- Avoiding or minimizing problematic areas mentioned in Section 5.1.3
- Testing impacts in the current system (e.g. failures, mistakes)
- Integration of the SLV Cement logistic system

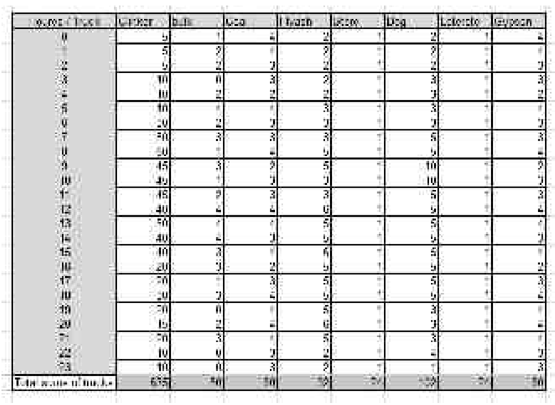
Logistic components include weighing systems in the entrance and exit gates, registering and managing customer orders and requirements, truck flow control, etc. By the **experiments** running in the built simulation models, it is possible to identify bottlenecks in the system, find a set of possible solutions and choose the optimal one for a given plant. The integration of SLV consists of loading the required data from the database, the implementation of the given plant configuration (number of components) and their properties.

5.1.5 Input data analysis

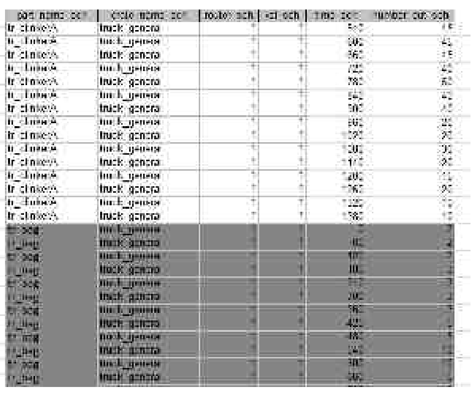
For a correct analysis and the building of an adequate simulation model, it is necessary to use valid data as:

- Definition of operations (inputs and outputs, operation times, etc.);
- Productive facilities and their properties (capacity, transportation speeds, breakdowns statistics, etc.);
- Layout specifications and graphical models for entities and facilities;

- Possible truck routings through production facilities and respective logic control;
- Workers and their properties (capabilities);
- Production schedule;
- Demand patterns (rate tables of arriving trucks);
- Patterns of failures;
- Shifts management.



Hours / Truck	Clinker	bulk	Coal	Flyash	Stone	Bag	Laterite	Gypsum
0	5	1	4	2	1	2	1	4
1	5	2	1	2	1	2	1	1
2	5	2	2	2	1	2	1	2
3	10	0	3	2	1	3	1	3
4	10	2	2	2	1	3	1	2
5	10	1	1	3	1	3	1	1
6	20	2	2	2	1	2	1	2
7	20	3	3	3	1	5	1	3
8	30	1	4	5	1	5	1	4
9	45	3	2	5	1	10	1	2
10	45	1	2	2	1	10	1	2
11	45	2	3	3	1	5	1	3
12	40	4	4	6	1	5	1	4
13	50	1	1	5	1	5	1	1
14	40	4	2	5	1	5	1	2
15	40	3	1	6	1	5	1	1
16	20	2	2	5	1	5	1	2
17	20	1	3	5	1	5	1	3
18	20	2	4	5	1	5	1	4
19	20	0	1	5	1	3	1	1
20	15	2	4	6	1	3	1	4
21	20	3	1	5	1	3	1	1
22	10	0	2	2	1	4	1	2
23	10	0	3	2	1	1	1	3
Total number of trucks	635	[illegible]	50	[illegible]	[illegible]	[illegible]	[illegible]	50



part_name_sch	crate_name_sch	router_sch	vol_sch	time_sch	number_out_sch
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_clinkerA	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]
tr_bag	truck_genera	1	1	[illegible]	[illegible]

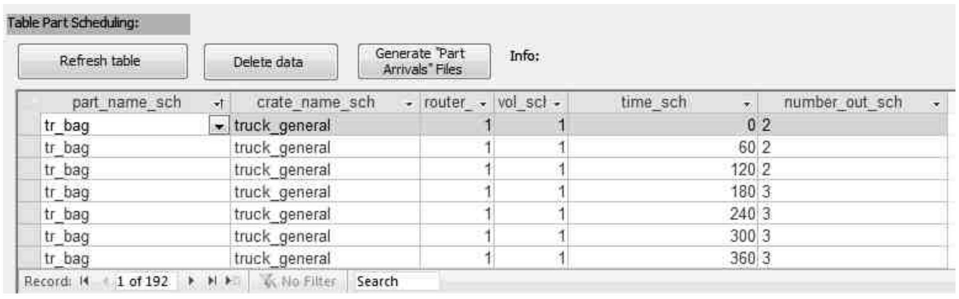


Table Part Scheduling:

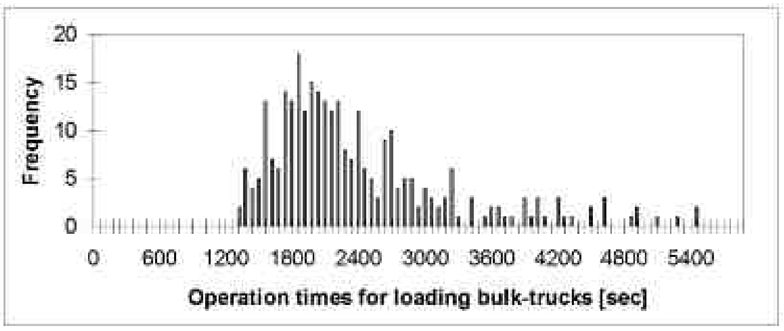
Refresh table | Delete data | Generate "Part Arrivals" Files | Info:

part_name_sch	crate_name_sch	router_	vol_sch	time_sch	number_out_sch
tr_bag	truck_general	1	1	0	2
tr_bag	truck_general	1	1	60	2
tr_bag	truck_general	1	1	120	2
tr_bag	truck_general	1	1	180	3
tr_bag	truck_general	1	1	240	3
tr_bag	truck_general	1	1	300	3
tr_bag	truck_general	1	1	360	3

Record: 1 of 192 No Filter Search

Figure 110 - Cachapuz: Data process of truck arrivals

These data can usually be obtained in internal logistic systems as ERP. In the case of the India plant, identical logistic system hasn´t been implemented there yet. Therefore, the input data used are of similar size and kind of the plant in Alexandria, Egypt. After the installation and implementation of the system, it is possible to use specific plant data. The rest of the data was received by a standard questionnaire. Figure 110 shows images from data processing of truck arrivals. The table with information about the trucks arrivals (number of the registered trucks) during the day (data from cement plant) is adjusted in MS Excel to an **IDS** format and data are copied to a DB table. Based on it, part files are generated as input for the simulation experiments. Figure 111 displays the analysis of the operation time with histograms to identify and parameterize appropriate stochastic functions (loading of trucks (bulk)).



	[sec]	[min]
Mean	2393.721	39.89536
Standard Error	54.53132	0.908855
Median	2124	35.4
Standard Deviation	941.3565	15.68927
Sample Variance	886152	
Kurtosis	16.04654	
Skewness	2.86835	
Range	8898	
Minimum	1285	
Maximum	10183	
Sum	713329	
Count	298	
Largest(1)	10183	
Smallest(1)	1285	
Confidence Level(95.0%)	107.3167	

Figure 111 - Cachapuz: establishment of the operation time

5.1.6 Model α – Simplified model

There are used two models with different **level of details** (Model α and β). This approach was chosen in order to make it easy to understand and establish the system configuration. In the first model (α), the system is simplified to see the essential behaviour and system objects. The second Model β, based on Model α, contains all the objects and describes the complete system behaviour.

Model α is a simplification of the real system (see Figure 112). Three basic kinds of trucks (bags, bulks, raw material) and six basic facilities (WBs, Warehouse, Silos, Raw material storages, parking queues and Check) are used. Some details of properties and functions as the WB logic, facility breakdowns and scheduling are not described.

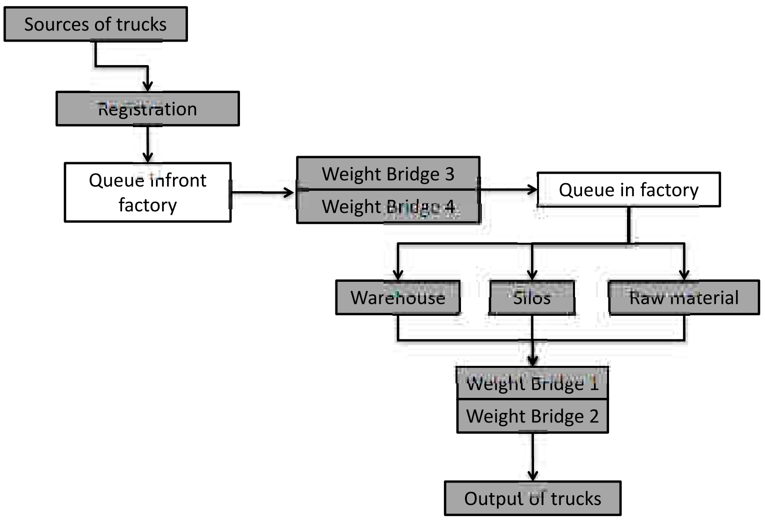
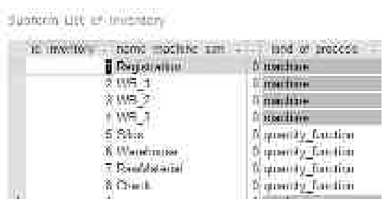


Figure 112 - Cachapuz: Schema of a simple simulation model α

This model consists of data processing (database), simulation model, result analysis and CAD layout.

Figure 113 consists of images from DB and generated simulation model. The results of the simulation experiment are in Figure 114. There are two graphs: one with the number of registered trucks in the parking place in front of the plant and the other with the number of trucks in the plant. According to the results, it is clear that the bottleneck of the system is in the Warehouse, as shown in the plot graphs "Number of trucks (bags) in the factory". These graphs also display that there are still almost 40 trucks in the factory and that limitation is to avoid that the factory is overcrowded.

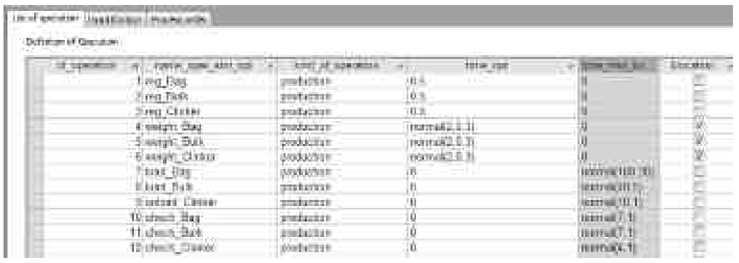
There are an increasing number of trucks (bags) in front of the plant, showing an unstable system. This model and the results help to understand the system and build a detailed Model β.



System List of inventory

id_inventory	name_machine_sim	kind_of_process
	1 Registration	0 machine
	2 WB_1	0 machine
	3 WB_2	0 machine
	4 WB_3	0 machine
	5 Silos	0 quantity_function
	6 Warehouse	0 quantity_function
	7 RawMaterial	0 quantity_function
	8 Check	0 quantity_function

List of resources

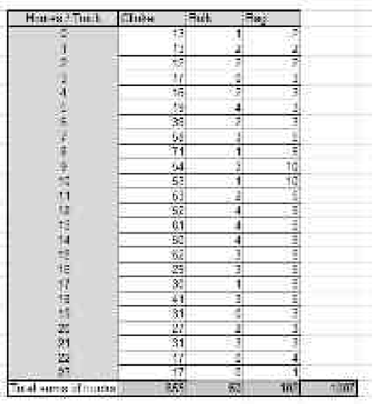


List of operation | [illegible] | [illegible]

Definition of Operation

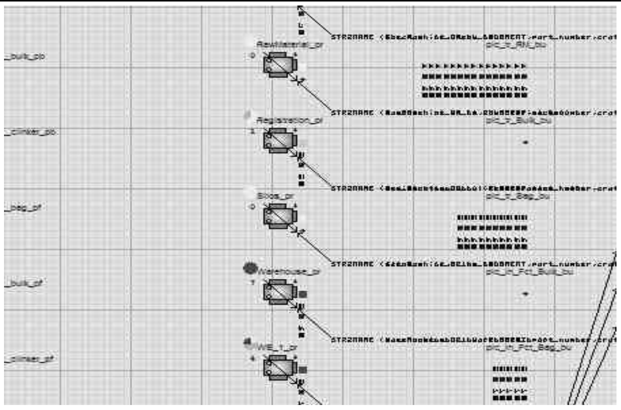
id_operation	name_operation_sim	kind_of_operation	time_op	time_machine	Duration
1	reg_Bag	production	0.5	0	
2	reg_Bulk	production	0.5	0	
3	reg_Clinker	production	0.5	0	
4	weigh_Bag	production	normal(2,1.1)	0	✓
5	weigh_Bulk	production	normal(2,0.1)	0	✓
6	weigh_Clinker	production	normal(2,0.1)	0	✓
7	load_Bag	production	0	normal(10,1)	
8	load_Bulk	production	0	normal(11,1)	
9	unload_Clinker	production	0	normal(10,1)	
10	check_Bag	production	0	normal(7,1)	
11	check_Bulk	production	0	normal(7,1)	
12	check_Clinker	production	0	normal(4,1)	

List of operation and operation times



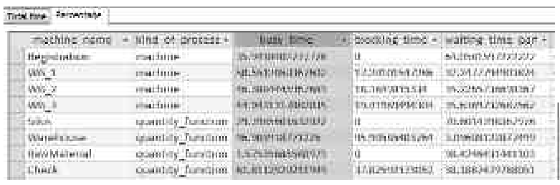
Hours\Truck	Clinker	Bulk	Bag
0	13	1	2
1	13	2	2
2	32	3	2
3	17	5	1
4	16	2	3
5	19	4	1
6	35	2	3
7	58	1	5
8	71	1	5
9	54	2	10
10	53	1	10
11	63	2	5
12	52	4	3
13	61	4	3
14	50	4	3
15	62	3	5
16	29	3	5
17	30	1	3
18	41	3	5
19	31	5	3
20	27	2	3
21	31	3	3
22	17	2	4
23	17	3	1
Total sums of trucks	855	62	107	1007

Arrival table



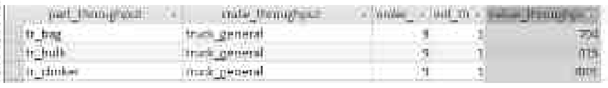
Generated simulation model

Figure 113 - Cachapuz: Model α - Input data overview and simulation model



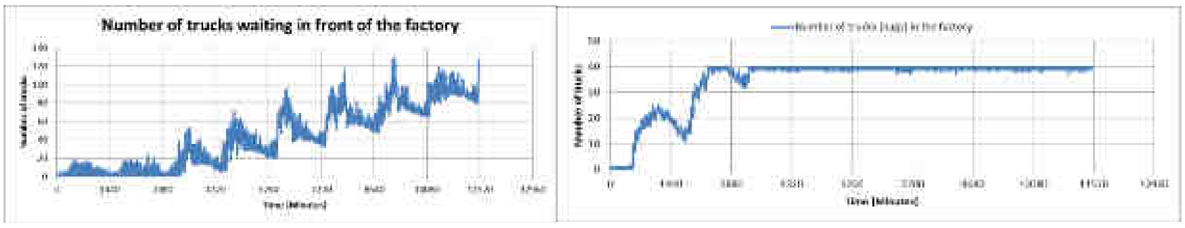
Total time | Percentage

machine_name	kind_of_process	busy_time	blocking_time	waiting_time_per
Registration	machine	[illegible]	0	[illegible]
WB_1	machine	[illegible]	[illegible]	[illegible]
WB_2	machine	[illegible]	[illegible]	[illegible]
WB_3	machine	[illegible]	[illegible]	[illegible]
Silos	quantity_function	[illegible]	0	[illegible]
Warehouse	quantity_function	[illegible]	[illegible]	[illegible]
RawMaterial	quantity_function	[illegible]	0	[illegible]
Check	quantity_function	[illegible]	[illegible]	[illegible]



part_throughput	mode_throughput	order_	unit_ti	value_throughput
fr_bag	truck_general	9	1	774
fr_bulk	truck_general	9	1	[illegible]
fr_clinker	truck_general	9	1	[illegible]

Tables of resource use and system throughput



Plots – number of trucks in the parking place in front of the plant and number of trucks in the plant

Figure 114 - Cachapuz: Model α – Results

5.1.7 Model β – Detailed simulation model

This model contains the detailed logic control with all the components (WB facilities, RM storages, etc.) and also with all types of trucks. In Figure 115, there is a schema of the system. The green colour represents the trucks (bags), the blue represents the trucks (raw material) and the black represents the trucks (coal); these names are used in DB and simulation model.

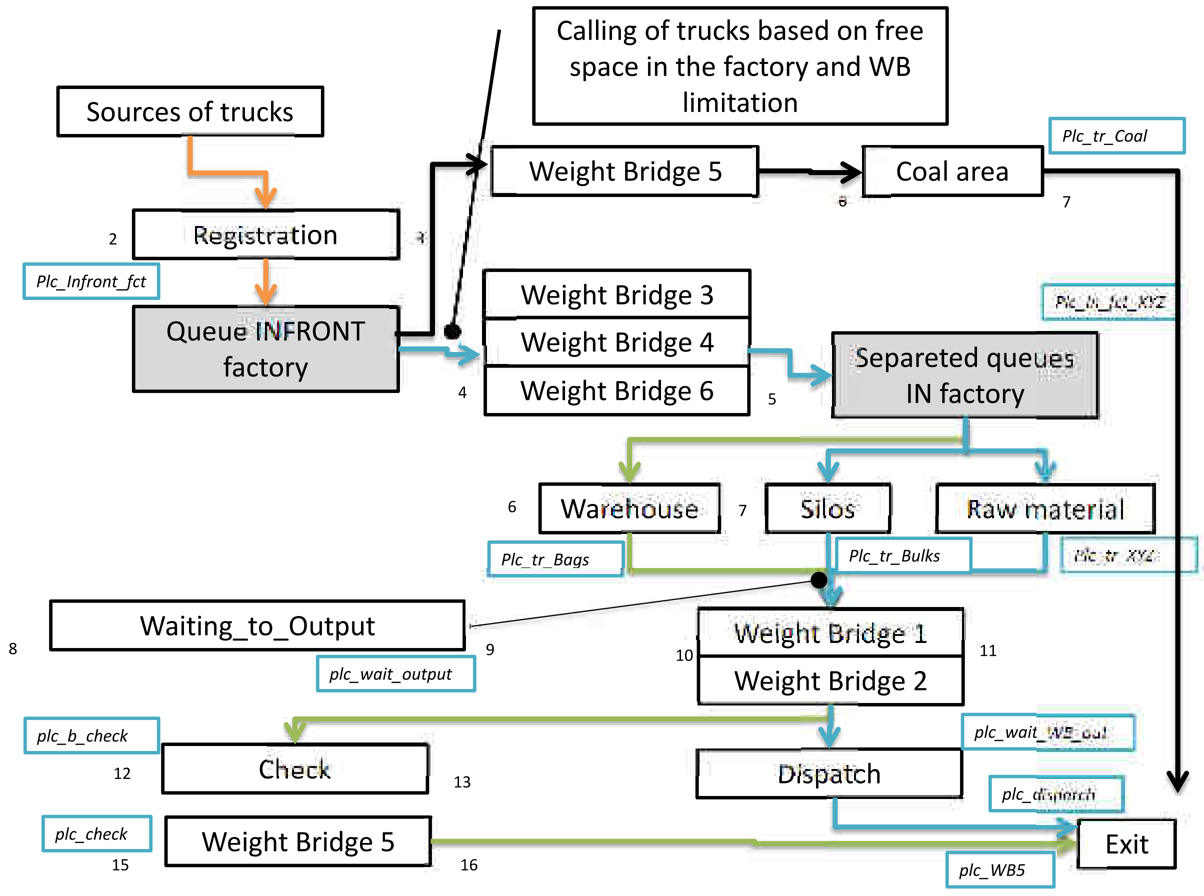


Figure 115 - Cachapuz: Schema of Model β

a) <u>Control logic</u>

Function "Calling trucks"

The trucks that are waiting in the parking areas in front of the plant are called to the plant by the display of information on the panel information board. Drivers can also receive text messages in their mobile phones when they are called to the plant (SLV implementation). The calling function is based on the FIFO principle for several different waiting trucks. If the first waiting truck cannot go forward due to the condition of limited number of trucks inside the plant, then the second (different kind) truck is called, etc. In **IDS**, it is directly implemented by pulling entities from buffer based on **different operation priorities**.

Limitation of the number of trucks

In order to avoid traffic jams inside the plant, a limitation function is used. That means that only a limited number of trucks can be inside the plant at the same time, while the rest must wait to be called. This is **managed by a buffer limitation and blocked property of operation in IDS**. These buffers represent all roads and spaces where trucks can stay in the plant.

WB logic control

WB in the plant contains specific logic control that selects trucks from the queue based on the conditions: kind of truck (WB just for trucks of raw material or trucks of bulk), weight limitation of WB and current state of the facility (if there is some failure, another gate should be chosen). The weight limitations of WBs are of 30, 60 and 100 tons. The implementation of this logic in **IDS** consists of the following data: **table of possible operation** with different priorities for each WB, **attributes for weight** and **pull system** from buffers.

The WB logic is based on accepting trucks with the appropriate attribute *vol*. This attribute is assigned based on the profile *"prf_Heavy_Loads"* that contains the distribution of trucks according to their weights (*vol* = 1 for trucks under 30 tons, *vol* = 2 for trucks under 60 tons, *vol* = 3 for trucks under 100 tons). The definition of the operation involves this attribute and the truck with appropriate weight for the given WB is chosen.

Mistakes

One of the very important issues is the influence of mistakes, especially in the loading operation. These mistakes can be caused by loading the wrong material or the wrong quantity (which is more often).

The **mistake** is assigned to the truck as an **attribute** based on the profile "*prf_Bag_Check*" (*vol=1* means that the truck had no mistake, *vol=2* means the opposite).

Trucks with loading mistakes are rejected in the dispatch area and **sent back** in order to re-check their loadings (Check and WB5). Afterwards, these trucks can leave the plant. In some cases, these mistakes could be critical as far as the flow of trucks in the plant is concerned. Moreover, it implies the re-use of resources to overcome the loading mistake.

b) <u>Simulation run and experiments</u>

Based on the input data and logic control, there is a generated simulation model (Model β). There are several experiments that are run to make the analysis of the system with different parameters, which are: warehouse loading positions (<u>warehouse capacity</u>), <u>operation time</u> and number of <u>arriving</u> trucks. The operation time is influenced by the implementation of the SLV logistic system (weight bridge operation and administration in dispatch) as well as by used equipment (e.g. manual loading and automatic loading system). Table 24 contains the **plan of experiments**.

In order to compare experiments results, it is used:

- Throughput of trucks (number of trucks that enter and leave the plant per unit time)
- Facilities utilisation;
- Waiting time of trucks for operation.

The length of the simulation experiment is 11520 minutes (8 days, which represent one week. The first day is used as warm-up period.

Table 24 – Cachapuz: Experiment overview

Name of experiment	Notice	Warehouse loading places	SLV implementation	Loading equipment	Trucks arrivals
Current state	Current state	10	No	Old	A
Experiment A	SLV	10	Yes	Old	A
Experiment B	Adding WHs	15	No	Old	A
Experiment C	Max throughput	10	Yes	New	B

Table 25 – Cachapuz: Operation times

Name of operation	SLV implementation	Loading equipment	Operation times [minutes]
Dispatch_Bag	No Yes	-	Normal (7,1) Normal (5,1)
Dispatch_Bulk	No Yes	-	Normal (5,1) Normal (3,1)
Dispatch_ClinkerA	No Yes	-	Normal (4,0.5) Normal (3,0.5)
Check_Bag	-	-	Normal (30,1)
Load_bag_A	-	Old New	Normal (160,30) Normal (60,30)
Load_Bulk	-	Old New	Normal (40,5) Normal (30,5)
Unload_RM	-	-	Normal (10,1)
Weight_Bag	No Yes	-	Normal (2,0.3) Normal (0.5,0.1)

c) Current system

This model's configuration represents the current system without the SLV Cement implementation. This system has the current machinery, **low level of automation** and the cement bags are loaded by using human power. The facilities have higher probability of breaks due to its age and state. The administration is based on paper documents usage; data are type in the ERP logistic system and then everything is printed and saved in the repository. The total number of **workers** in logistic per shift is **27**.

These graphs display an unstable system (see Figure 116). In this case, it represents a very long waiting time that corresponds to the real situation, overcrowding the parking space in front of the plant. In the real system, the waiting queue of a registered truck does not grow indefinitely. In the model, that happens due to conditions that are not specified by the company – as extra work time of staff, not precised arrival table or customers that cancelled their order. This difference can even be of 10 trucks per day.

The second graph shows that the number of trucks (bags) in the plant is almost 40, which is the established limit. The most used facilities are the Warehouse and the Dispatch.

d) Experiment A

This experiment describes a system after the implementation of logistic components (traffic control) connected with the SLV Cement logistic system. This upgrade basically means **shorter operation time** (weighing is only 30 seconds), administration (dispatch), etc. and mainly **saving staff** at the WB (7 workers per shift). Now truck drivers are using an electronic access card and the operation is completely self-service. The use of WBs duly decreases the markedly shorter weighing times.

The total plant throughtput compared with the current states is almost the same. Warehouse is the bottleneck of trucks (bags) (see Figure 117).

The experiment A shows that only the SLV installation does not increase throughput. However, it saves 7 members of the staff in WBs, and decreases the average waiting time for the other kind of trucks (raw material), with the exception of the truck (bags), as shown in the graph.

Due to implementation of SLV logic control for traffic, trucks (raw material) have a higher priority than trucks (bags), so trucks (raw material) don't have to wait in the queue in front of the plant.

e) Experiment B

This experiment shows the behaviour of the plant with **15 loading positions** for truck with bags in the Warehouse (increasing 5 positions to avoid the bottleneck), keeping old WBs and any IT solution (SLV).

The total number of **staff is 32** (5 more than the warehouse). The system is stable and the total throughput is the same as in the current plant state or in the Experiment "A" (Figure 118). The waiting time in front of the plant is around 35 minutes shorter now.

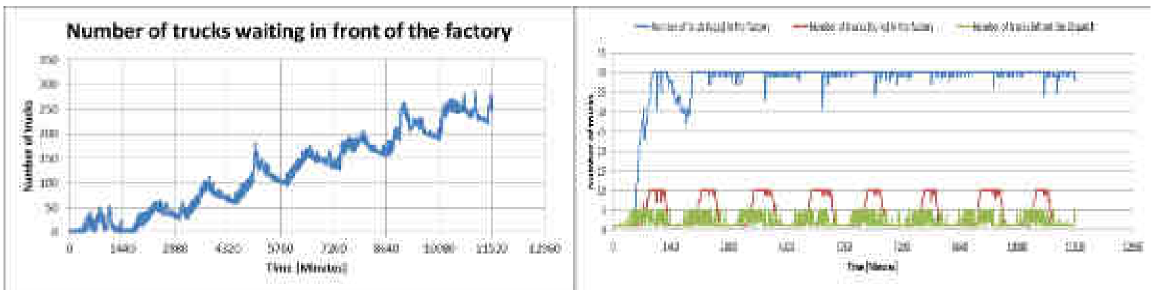
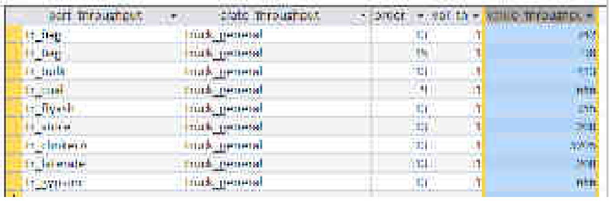


Figure 116 - Cachapuz: Current state results



part_throughput	state_throughput	order	vol_ta	value_throughput
tr_bag	truck_general	11	1	[illegible]
tr_bag	truck_general	15	1	[illegible]
tr_bulk	truck_general	11	1	[illegible]
tr_coal	truck_general	4	1	[illegible]
tr_flyash	truck_general	11	1	[illegible]
tr_stone	truck_general	11	1	[illegible]
tr_clinkerA	truck_general	11	1	[illegible]
tr_laterite	truck_general	11	1	[illegible]
tr_gypsum	truck_general	11	1	[illegible]

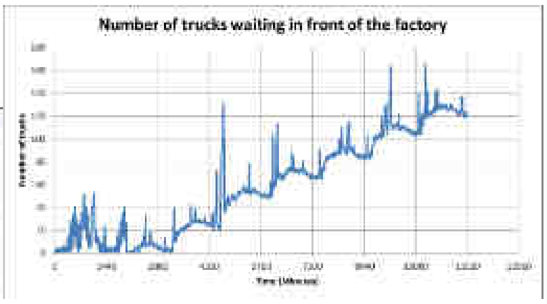


Figure 117 - Cachapuz: Experiment “A” results

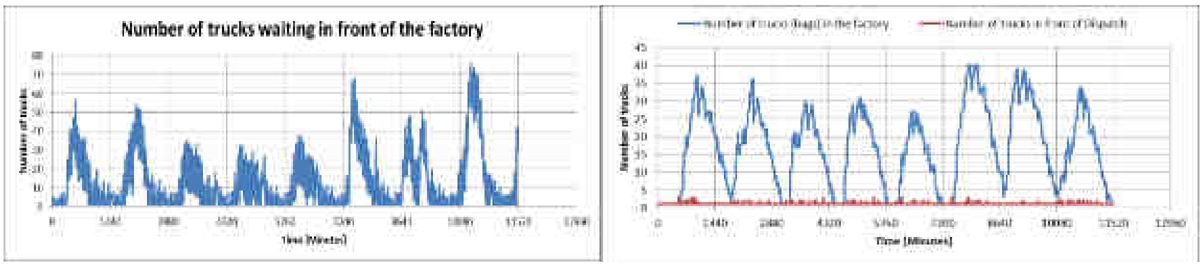
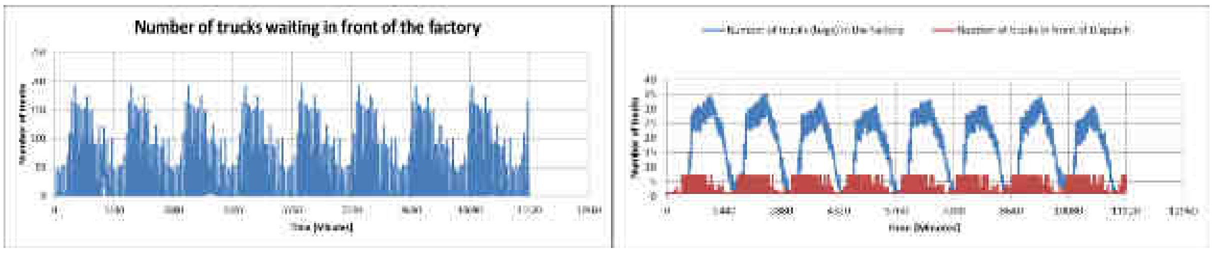
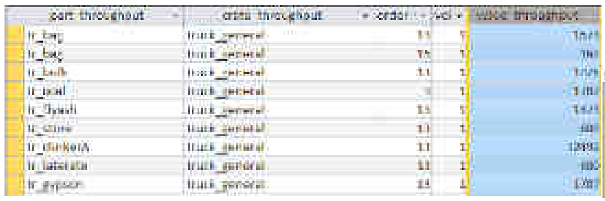
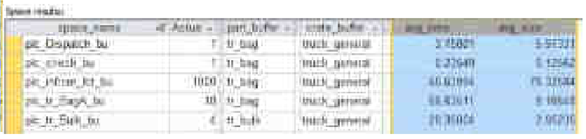


Figure 118 - Cachapuz: Experiment B results





part_throughput	state_throughput	order	vol	value_throughput
tr_bag	truck_general	11	1	[illegible]
tr_bag	truck_general	15	1	[illegible]
tr_bulk	truck_general	11	1	[illegible]
tr_coal	truck_general	[illegible]	1	[illegible]
tr_flyash	truck_general	11	1	[illegible]
tr_stone	truck_general	11	1	[illegible]
tr_clinkerA	truck_general	11	1	12892
tr_laterite	truck_general	11	1	880
tr_gypsum	truck_general	14	1	1287



Space results:

space_name	# Active	part_buffer	state_buffer	avg_time	avg_size
pic_Dispatch_bu	7	tr_bag	truck_general	[illegible]	[illegible]
pic_check_bu	7	tr_bag	truck_general	[illegible]	[illegible]
pic_Inflow_lct_bu	1000	tr_bag	truck_general	[illegible]	[illegible]
pic_tr_BagA_bu	10	tr_bag	truck_general	[illegible]	[illegible]
pic_tr_Bulk_bu	4	tr_bulk	truck_general	[illegible]	[illegible]

Figure 119 - Cachapuz: Experiment C results

f) Experiment C

This experiment involved the implementation of the **SLV system** and the **upgrade** of the facilities as the **automation of loading bags** to trucks. A different set of arrival tables that contain **more incoming trucks** is used. The number of arriving trucks is multiplied by the coefficient 2. It is based on the basic capacity calculation for a daily throughput of truck (bags) through the Warehouse as bottleneck, 1440 * 10(loading places) / 60 min = 240 truck (bags) per day in a 100% use. The total number of **staff** per shift is **22**. It is needed two extra workers in the dispatch. Figure 119 shows the results: a stable system and doubled throughput of truck (bags) customers.

g) Results of experiments

Based on the results from the current state and experiments, it is clear that the bottleneck in the system for trucks (bags) is the Warehouse. The current number of WBs seems to be adequate and it does not influence the global system throughput.

It was previously thought that, due to the high number of trucks waiting in the entrance gate, the number of gates would be critical to the overall system performance. Instead, it is now clear that the main problem is concerned with the number of loading places in the warehouse – in fact these numbers of loading places affect the system performance.

There are basically two possibilities to increase the system throughput and decrease the waiting time – adding more loading places in the Warehouse (Experiment B for 15 loading places) or implementing automatic and faster loading equipment (Experiment A and C).

With the installation of the **SLV logistic system**, the waiting time of trucks (raw material) is decreased (Experiment A) because the weight operations and administration in the dispatch take less time. It also saves staff in the WBs (in this case 7 work positions per shift).

The highest performance configuration of the system can be a combination of updated equipment with the SLV system (Experiment C). The total throughput is increased from around 9000 to 21500. In general, around 2.2 times more trucks (customers - bags and bulks) and suppliers (trucks (raw material)) and even 5 working positions are saved (27→22).

These experiments can also be a support for the cement plant management to decide to make an investment. An important issue that influences investment is the low salary for manual work in India.

5.1.8 CAD layout

a) Layout and material flows

In this example, the CAD layout helps to display material flows and create 3D models. The plant layout was sent from a cement company only as a schema without any scale and in image format (Figure 120). Therefore, *Google Earth Application* is used. It is possible to use it because of the large size and good map resolution of outdoor facilities, so weight bridges with trucks in the queues can be identified. This image is used as a background in the AutoCAD drawing. The blocks representing resources (WB, Warehouse, Silos, etc.) are generated by **IDS** to this layout and arranged in the correct position, based on the plant image in the background. This image also helps to draw a network of roads. Then, the material flows can be generated by IDS, as shown in Figure 121. There are schematic flows (straight) between resources and for specific truck (bags) or trafic flows based on road network. It helps the optimal installation of the SLV logic control system for traffic.

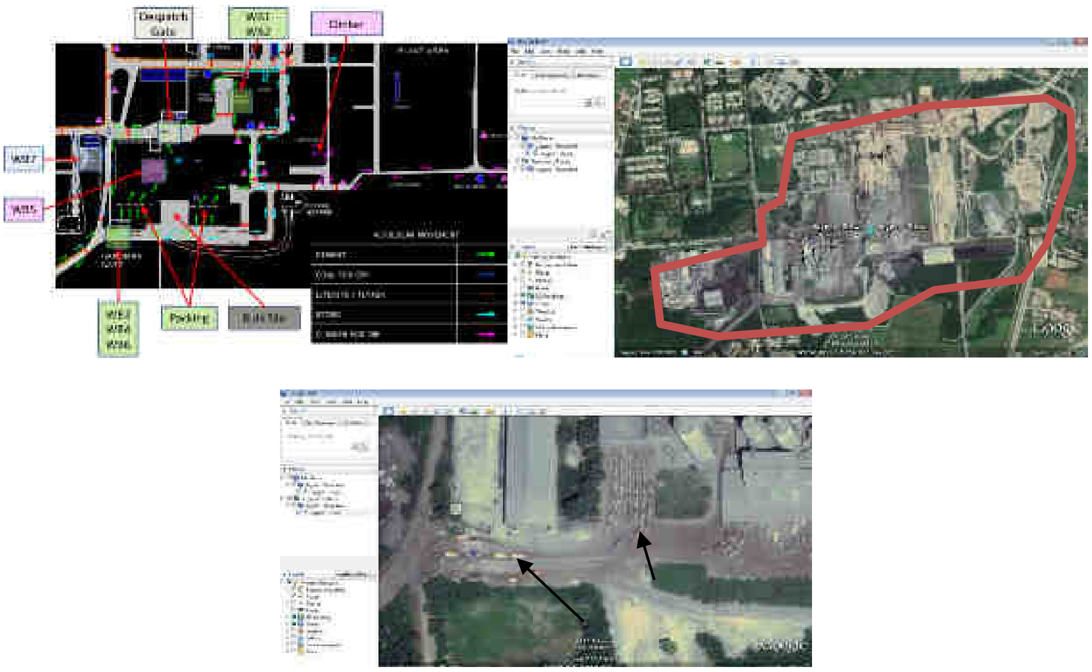


Figure 120 - Cachapuz: CAD layout data processing (source: www.jalindia.com, 2011)

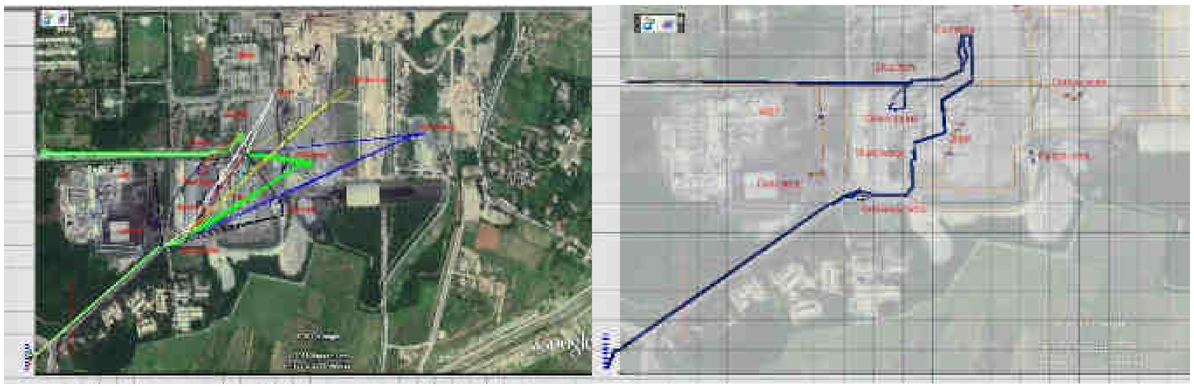


Figure 121 - Cachapuz: CAD layout – material flows

Figure 122 - Google 3D Warehouse

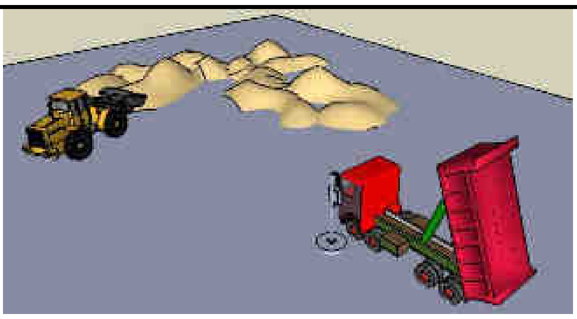
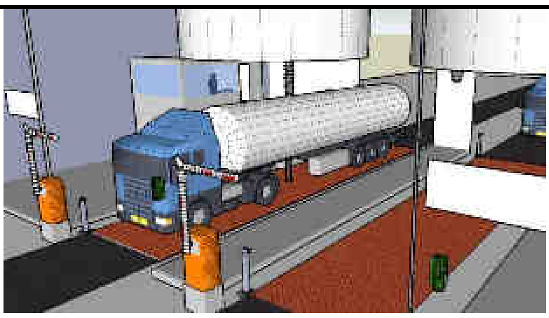
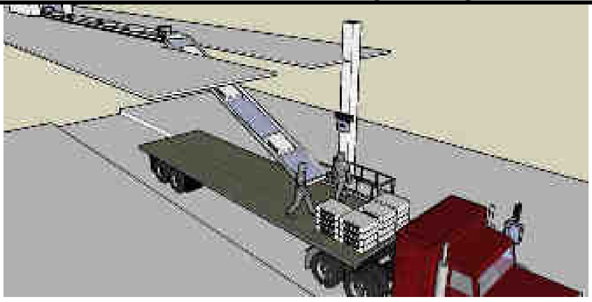
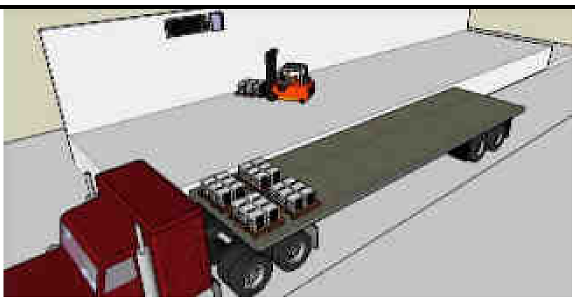
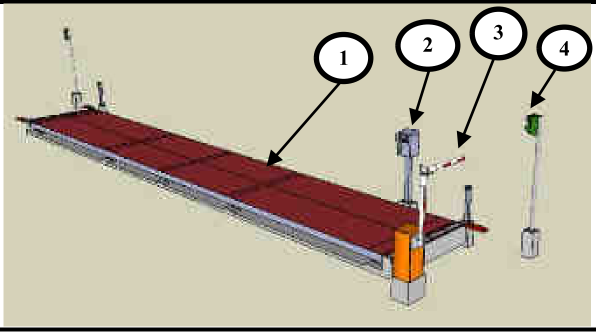
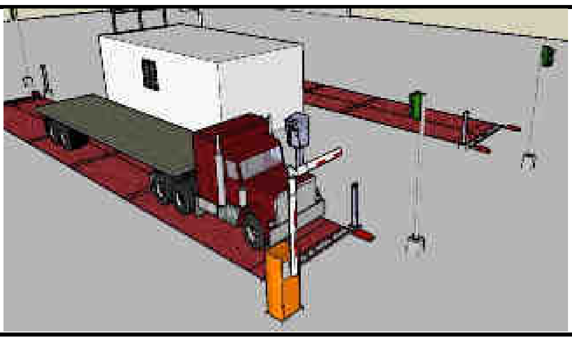
 **Raw material unloading**	 **truck (bulk) loading in the Silos**
Three levels of automation of bag loading	
 **Manual loading**	 **Forklift loading**
Pneumatic loading of bags	**Weight bridges and logistic control elements:** 1. Weight bridge platform 2. Terminal with RFID card reader 3. Security gate 4. Traffic light
	

Figure 123 - Cachapuz: 3D drawing models of cement plant elements

b) <u>3D model</u>

In order to create a realistic 3D model in AutoCAD, it is possible to use free drawing models from the internet application called *Google 3D Warehouse* (see Figure 122) or models from similar factories. These models can be changed and adjusted in the *Google SchetchUp* modelling application. After exporting to the AutoCAD format (*"dwg"* or *"dfx"*), they can be placed in the plant layout, as shown in Figure 123. There are illustrated images of facilities equipment from the cement plant of Alexandria, Egypt. The 3D models were created in Cachapuz Company.

5.1.9 Overall results and comparison of two approaches

The use of an internal logistic system and DES for the design of this type of plant seems to be a good approach. It is possible to implement virtually a **logistic control system** to an existing plant and analyse the corresponding impact without any type of physical intervention in the real plant. The optimal system configuration can be found for each set of factories resources (e.g. facilities, space, human resources). The simulation could also be used for:

- Testing impacts of failures, breakdowns or crashes and predicting ways of overcoming these situations
- Avoiding bad design decisions through previous tests with the simulation model
- Testing system stability with different scenarios
- Implementing stochastic influences into the design of the system
- Reducing customer (trucks) waiting time

Comparing to the project done in the simulation tool SIMIO, results of both projects are very similar: both models give the same information but the differences are in the time requirement. Cachapuz project in SIMIO needed around 6 weeks of work, the most time-consuming is creating of simulation model. IDS project needed only 3 weeks. Most of the time - 2 weeks have been used for input data preparation and analysis. Complete estimated values are in Table 34.

The model done in SIMIO was also built to help sell the SLV system (promotion purpose), therefore a 3D realistic graphic is developed. The snapshots are in the Appendix (Section A.5.1), 3D videos reference (www.youtube.com/watch?v=GLlbof30FxE, 2010), the process of creation is described in a book chapter (Dias, Vik, Pereira & Oliveira, 2012).

5.2 Blaupunkt Bosch

5.2.1 Project overview

This project describes an internal logistic system, including the use of milk-runs (MR), also called Logistic Trains (LT), in the electronic industry, more precisely in the company Blaupunkt Braga Ltd., production of car radios and GPS navigators (www.bosch.pt, 2011).

With the IDS solution, the MR design and process analysis are performed with computer simulation, DB and CAD. In this example, it is shown the integration and feedbacks in the design.

The project outcomes, based on the simulation optimization, lead to the reduction of logistic costs and improvement in the availability of material in production lines. These in turn, reduce work in process material (by reducing stock levels and buffers in the production lines) and setting the minimum number of MR.

This system was studied in a previous project (2007-8) with the simulation tool Arena and 3D Arena Player, which is used as the visualisation tool. It was focused on realistic 3D graphics and illustrative animation to describe the kanban system and MR principle. Time and data requirements of both approaches (Arena and IDS) are also compared (Vik, Dias, Oliveira, & Pereira, 2008a)(Vik, Dias, & Oliveira, 2008b)(Dias, Vik, Pereira & Oliveira, 2012).

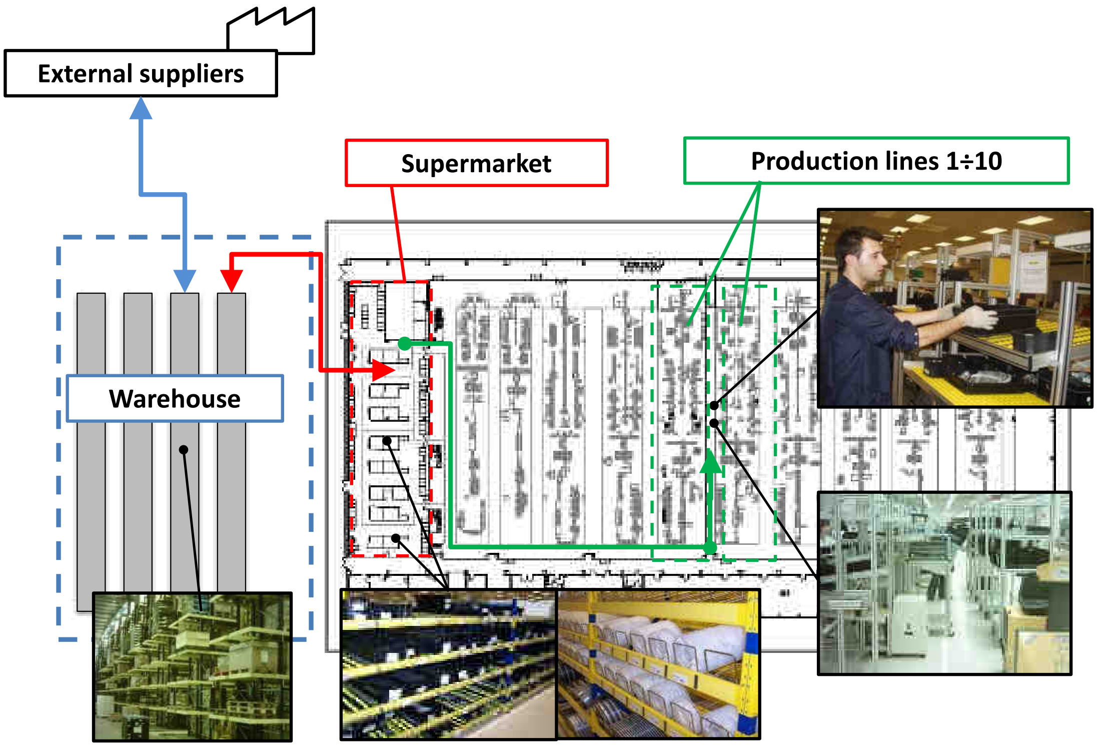


Figure 124 - Schema of internal logistic system

5.2.2 Description of the project and task definition

The principle of supplying with MR is described in Section 4.7.2 and the IDS implementation in Section A.4.11. The main principle of MR is supplying goods from deposits to delivery points in fixed and defined routes and cycle times, and also picking empty transportation units. It is more effective and better control if the kanban system implementation.

Figure 124 displays a schema of complete supplying processes in the factory. There are three supplying levels. Electronic components are supplied by external suppliers in no-regular cycles (days, weeks) with large packages of goods (e.g. pallets). These items are stored in the ***Warehouse***, and ***Supermarket*** is supplied by those items based on kanban orders. ***Production lines*** are supplied by a milk-run system in periodic cycles. ***Production lines*** consist of stations that are places with buffers. There are 10 production lines.

Project tasks:

The MR supplying system is defined by the **number of MRs** units, the **cycle time** of MR and **scheduling** the arrival, **routes** and **stations**. Based on the structure of the delivered items and the consumption of production lines, the supplying politics are chosen. In this case, it is a kanban system that is established by the appropriate **number of kanban cards** or transportation units in the supplying system.

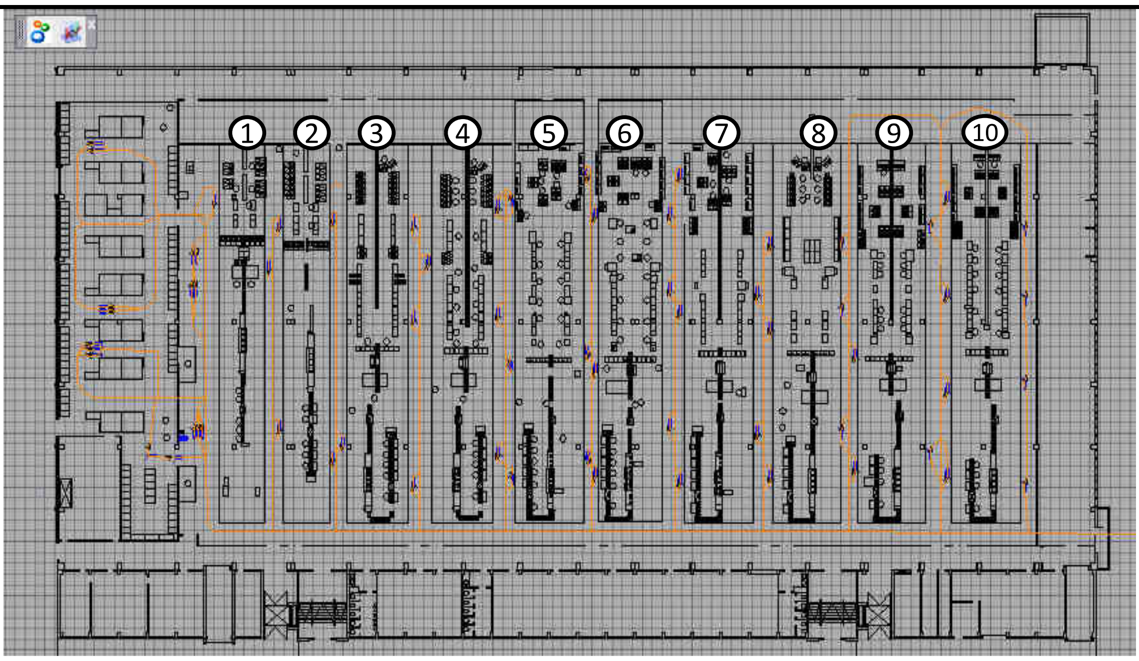
5.2.3 Input data processing

Input data consists of a factory CAD layout with a defined transportation network, observable in orange colour in the first image of Figure 125. Based on the layout, routes and delivering stations are defined for each MR. Stations are loading and unloading in the *Supermarket* and near the production lines. The MR transporter is defined by the transportation speed, turning diameter (manoeuverability) and capacity (number of wagons).

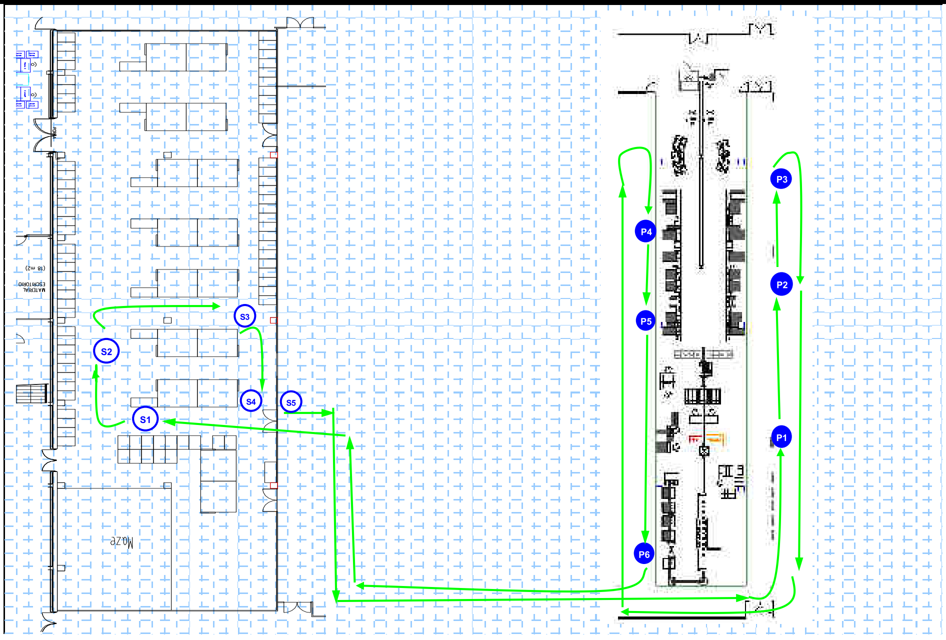
5.2.4 Design of a delivering system in IDS

Designing a system with good performance, considering all constrains, is a complex mission. It is necessary to consider space factory limitation, inventory (equipment), schedules, transported volumes, etc. to achieve an optimal and flexible solution. IDS supports all mentioned issues by built-in functionalities:

- automatic generation of a simulation model
- milk-run transportation system
- generation of CAD layouts and material flows
- shortest path searching and routes design

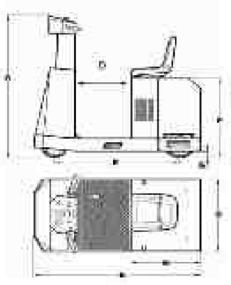


Factory layout and transportation network



Definition of routes and delivering stations

Autonomia (approssimativa) Autonomy (approx.) - Autonomie (approximative) Autonomie ca. - Autonomía (aproximada)	Traction 8 h Super traction 12 h Gel 8 h
Velocità massima Maximum speed - Vitesse maxi Höchestgeschwindigkeit - Velocidad máxima	1:11 10.5 km/h 1:16 8 km/h 1:22 5 km/h
Portata (compreso guidatore) Weight bearing capacity (including driver) - Portée (opérateur compris) Tragfähigkeit (einschließlich Fahrer) - Capacidad (incluido el conductor)	200 Kg
Traino su piano Towing capacity - Traction Schleppfaehigkeit - Remolque	1500 Kg
Peso (escluso batterie) Weight (without batteries) - Poids Gewicht (ohne Batterie) - Peso	193 Kg
Tipo di motorizzazione Type of drive - Type de motorisation Motorisierung - Tipo de motorización	**Elettrica** Electric - Électrique Elektrisch - Eléctrica



MR properties

Figure 125 - Inputs for MR project

Based on the project tasks, it is possible to simplify the solution and divide it into **two models** with different level of details and different insights. These two models need some type of integration, where information and data are shared.

Figure 126 shows data interconnection of the two models (α and β). Together, they describe completely the behaviour of this internal logistic system. Data are shared between model α, where the CAD **layout** is more important, and model β, whose focus is concentrated on the process of production management policy – **kanban** setup.

This is also one of the general benefits of IDS – sharing information between less complex models.

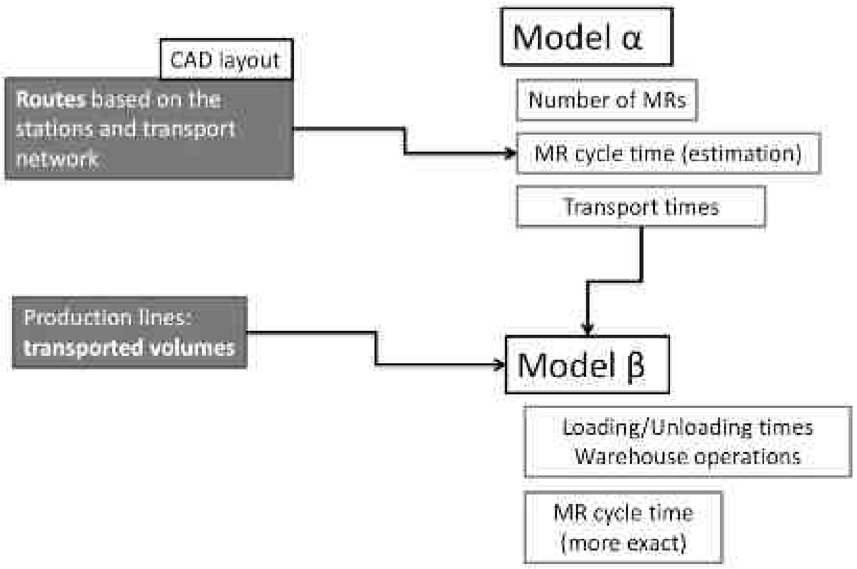

CAD layout
Routes based on the stations and transport network
Model α
Number of MRs
MR cycle time (estimation)
Transport times
Production lines: transported volumes
Model β
Loading/Unloading times Warehouse operations
MR cycle time (more exact)

Figure 126 - Data-merging of simulation models

5.2.5 Model α – 10 MRs

Initially, it is considered 10 MRs, each supplying one production line. Each MR is represented by an entity with defined routes. Route is a sequence of processes with transportation times as operation time. **Transportation times** are calculated automatically by **IDS** based on the speed and transportation distances determined in the CAD layout. This model does not include exact loading/unloading volumes (number of boxes) but only times for these operations (see model β). The main aim is to design transportation routes and the required MR cycle time.

Outputs from model α are:

- Routes design
- Use of MRs
- Traffic accumulated frequency in the main roads (shared by several MRs)

Simulation model description

A MR cycle consists of transportation time, loading/unloading items at the stations, loading/unloading items in the Supermarket and waiting time until the end of the MR cycle period. The setup of the operation time is shown in Figure 127 (upper image).

In order to describe transportation activities, *"time path"* property is used. Initially, the transportation time is 10 seconds based on the default length between stations of 20 metres, and the default speed of 2 metres per second (see Figure 169, lower image).

Loading/unloading time at the stations is 45 seconds as default value. The supermarket operation (changing boxes in the supermarket) takes 45 seconds * number of stations, e.g. MR#5 needs 45 * 6 = 270 seconds.

The waiting time of MR is given as:

```
(MR_cycle_time)* function(number of operation)-TIME

waiting time= 1200 * P_2_tt(P_2_pa,3)-TIME
```

where *function(number of operation)* represents the total number of performed cycles(laps) including the current run ("P_2_tt(P_2_pa,3)"), and *TIME* is the simulation run time. 1200 is the cycle time in seconds.

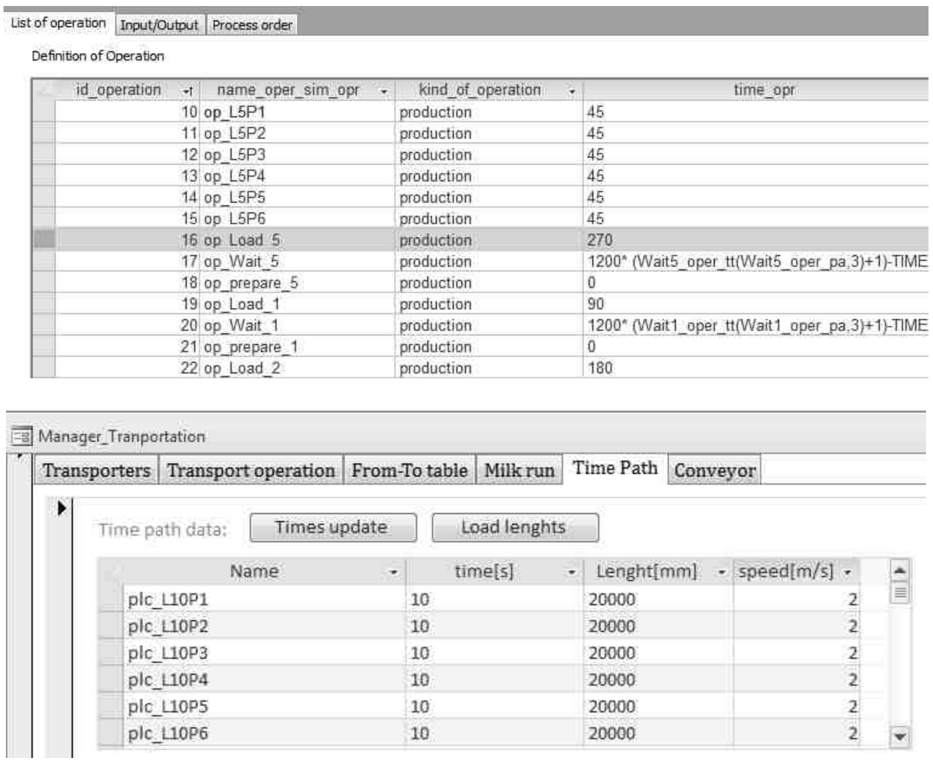
After preparing data, the simulation model can be generated and the first results from the simulation model are received.

In this phase, results as values are not so important because there are no accurate data yet (only default). Accurate data, as transportation distances, can be automatically calculated from the CAD layout, by **IDS**.

CAD layout

The design of the MR system in CAD is processed by generating blocks that represent resources in the factory layout and placing them into positions (stations) near the production lines. After that, the transport flows can be automatically generated by the function "**Shortest path**" that searches the shortest path in the transportation network, as it is shown in Figure 129. In the upper image, there are routes of two MRs: Milkrun#5 (red colour) and Milkrun#3 (blue colour).

The **transportation movement frequency** is a very important information. It gives the number of movements in each section of the transportation net. Figure 129 (lower image) shows that the highest values are near to the Warehouse (around 1152 movements per day, considering the sum of both directions).



List of operation | Input/Output | Process order

Definition of Operation

id_operation	name_oper_sim_opr	kind_of_operation	time_opr
10	op_L5P1	production	45
11	op_L5P2	production	45
12	op_L5P3	production	45
13	op_L5P4	production	45
14	op_L5P5	production	45
15	op_L5P6	production	45
16	op_Load_5	production	270
17	op_Wait_5	production	1200* (Wait5_oper_tt(Wait5_oper_pa,3)+1)-TIME
18	op_prepare_5	production	0
19	op_Load_1	production	90
20	op_Wait_1	production	1200* (Wait1_oper_tt(Wait1_oper_pa,3)+1)-TIME
21	op_prepare_1	production	0
22	op_Load_2	production	180

Manager_Tranportation

Transporters | Transport operation | From-To table | Milk run | Time Path | Conveyor

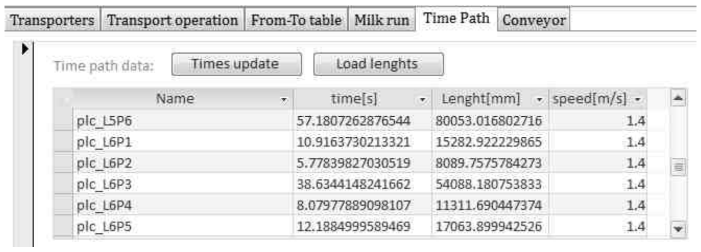
Time path data: Times update | Load lenghts

Name	time[s]	Lenght[mm]	speed[m/s]
plc_L10P1	10	20000	2
plc_L10P2	10	20000	2
plc_L10P3	10	20000	2
plc_L10P4	10	20000	2
plc_L10P5	10	20000	2
plc_L10P6	10	20000	2

Figure 127 - Model α: Setup of operation times and Transportation setup

Results and outcomes from CAD and simulation

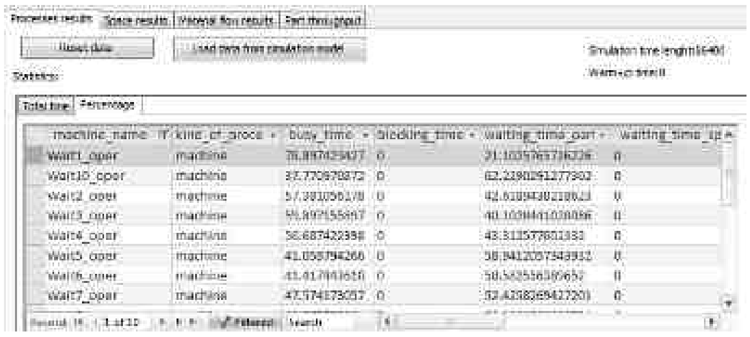
The next step is to save route information (length of each transportation route section) into DB and recalculate the transported time. The speed is changed to actual 1.4 metres per second (see Figure 128, upper image). After updating these values, a new simulation experiment can be processed and accurate results can be analysed (see Figure 128, lower image).



Transporters | Transport operation | From-To table | Milk run | Time Path | Conveyor

Time path data: Times update | Load lenghts

Name	time[s]	Lenght[mm]	speed[m/s]
plc_L5P6	57.1807262876544	80053.016802716	1.4
plc_L6P1	10.9163730213321	15282.922229865	1.4
plc_L6P2	5.77839827030519	8089.7575784273	1.4
plc_L6P3	38.6344148241662	54088.180753833	1.4
plc_L6P4	8.07977889098107	11311.690447374	1.4
plc_L6P5	12.1884999589469	17063.899942526	1.4



Processes results | Space results | Material flow results | Part throughput

Reset data | Load data from simulation model

Simulation time length 16400

Warm-up time 0

Statistics:

Total time | Percentage

machine_name	kind_of_proce	busy_time	blocking_time	waiting_time_part	waiting_time_tp
Wait1_oper	machine	78.897425927	0	21.1025765736226	0
Wait10_oper	machine	37.770970872	0	62.2290291277302	0
Wait2_oper	machine	57.381056178	0	42.6189438218823	0
Wait3_oper	machine	59.897155897	0	40.1028441020086	0
Wait4_oper	machine	56.687422388	0	43.3125777601333	0
Wait5_oper	machine	41.058794268	0	58.9412057343932	0
Wait6_oper	machine	41.417463650	0	58.5825536350657	0
Wait7_oper	machine	47.574173057	0	52.4258269427201	0

Record: 1 of 10 | Filtered | Search

Figure 128 - Model α: Transportation time and Results

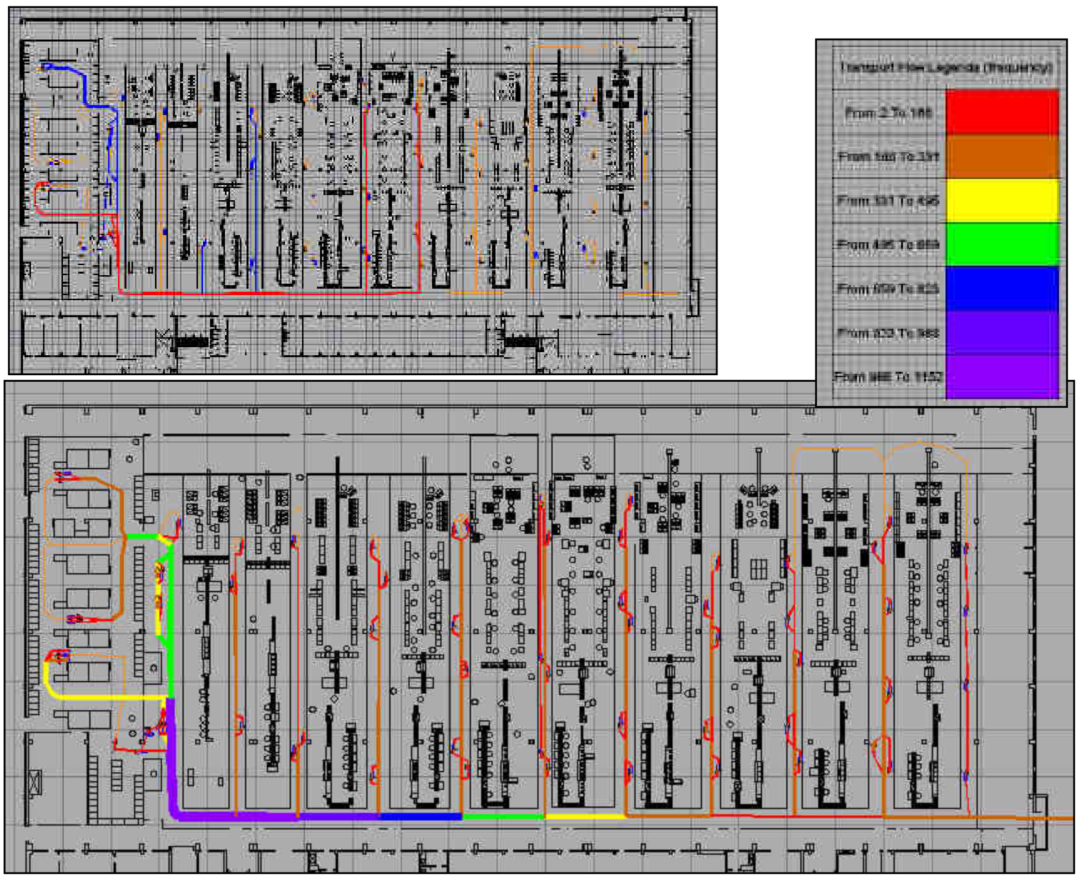


Figure 129 - Model α: layouts

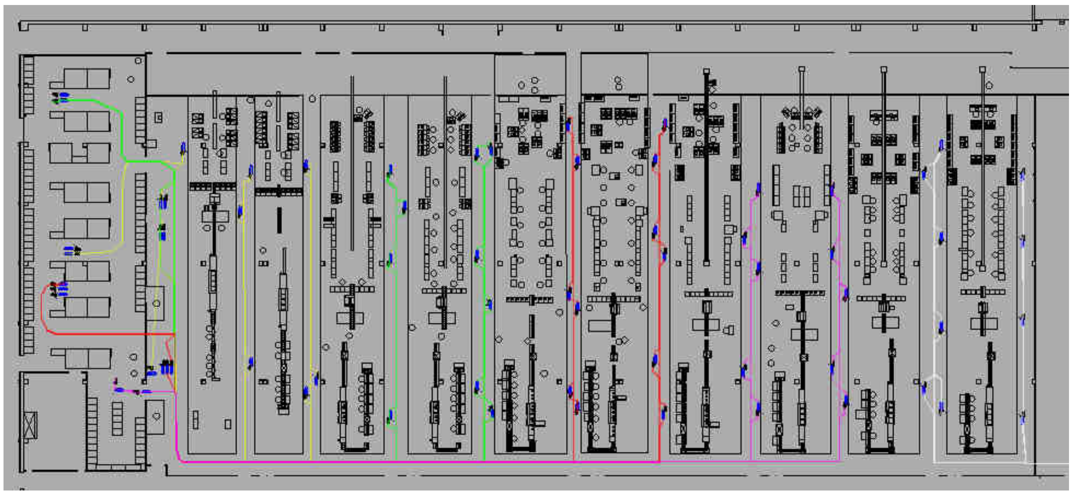


Figure 130 - Model α: Experiment “A” – 5 MR

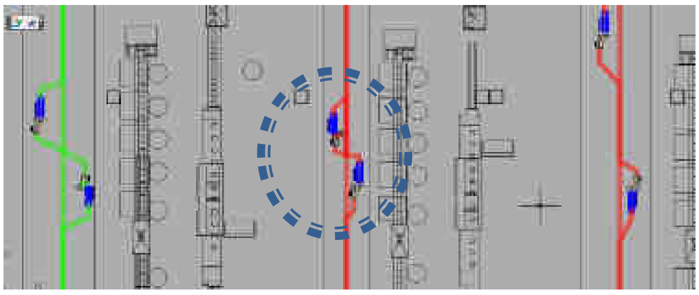


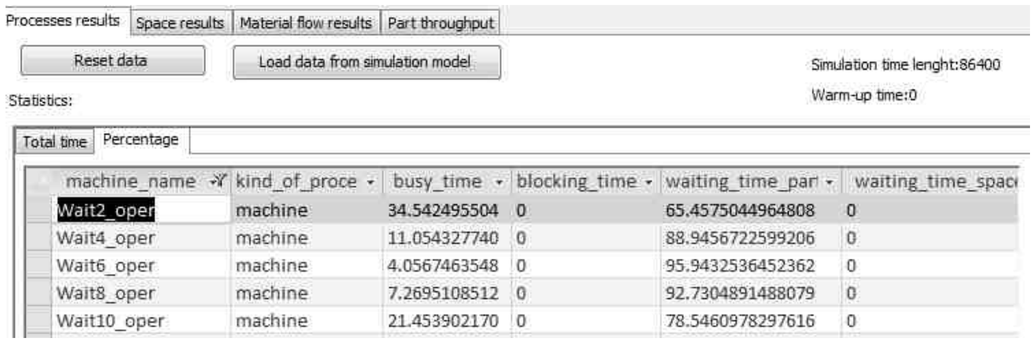
Figure 131 - Model α: Experiment “A” – Joining of stations

Explanation of the states meaning:

MR must stay in the Supermarket until the beginning of a new cycle, in this case, the rest of the time up to 1200 seconds. This is managed by **wait_machines** process. It is important to explain that for instance, for the MR#5, the process that makes the MR to wait is named **Wait5_oper**. The value of "busy_time" of that process is percentage of time that the MR#5 is waiting (41%). The value of "waiting time" is the percentage of time that the MR#5, is moving around its cycle (59%).

a) Experiment A (Model α)

Based on the results from Figure 128, the average use of each MR is around 50%. This experiment tries to decrease the number of MR from 10 to 5. One MR supplies two production lines, as shown in Figure 130. There is a green route for MR#4 supplying, on the left side of #3, in the complete #4 and on the right side of #5 production lines. The red colour is for MR#6 that also supplies all stations in two aisles.



Simulation time lenght:86400

Warm-up time:0

Statistics:

machine_name	kind_of_proce	busy_time	blocking_time	waiting_time_pari	waiting_time_spac
Wait2_oper	machine	34.542495504	0	65.4575044964808	0
Wait4_oper	machine	11.054327740	0	88.9456722599206	0
Wait6_oper	machine	4.0567463548	0	95.9432536452362	0
Wait8_oper	machine	7.2695108512	0	92.7304891488079	0
Wait10_oper	machine	21.453902170	0	78.5460978297616	0

Figure 132 - Model α: Experiment "A" – Results

During the validation in the simulation model, a message appears noticing that the cycle time of MR#6 is too short for managing all transportation and loading operations and that it needs extra 136 seconds. One possible solution can be to join two stations (see layout Figure 131).

Some of the stations of both sides of the aisle are very close to each other (a few metres), so it is possible to join them into one station and reduce loading/unloading time.

The results of this experiment confirm that it is **theoretically possible** to reduce the number of MRs from 10 to 5, so MRs have enough time (transportation and loading) to manage the complete cycle (image of Figure 132).

Results and discussion about Model α and experiments

It is possible to reduce the number of MR to 5 as well as MR drivers but after discussion amongst the team, the main decision is to reduce only to 9 (by one) because of safety reasons.

Each production line is focused on specific customers and it is not appropriate to mix them together.

This kind of supplying is quite critical for production, so breakdowns or failures (human) can stop the production.

However, it was rejected in the management meeting due to the **big risk** and also the mixing parts of more customers together. By risk, it means that the waiting time also covers safety time for solving problems and other stochastic influences that are not considered. In this case, this safety time is very short.

Other counterargument is that these MRs will take more boxes, increasing the number of wagons. It can cause a problem of turning MR in narrow aisles between production lines (problems of manoeuvrability).

This reduction will only be used to reduce MR#1 and MR#2 where free time is available. Some of the MRs needs less time because of the shorter routes, but due to uniformity, a common time is established for all of them.

5.2.6 Model β – 1MR

This model contains more details describing MR behaviour, as loading/unloading volumes and kind of items. It was decided to simulate this detail level to just one MR, because they all have an analogue behaviour and they are **independent** of each other. Routes and supermarket are not occupied (low traffic) and there is small risk of some kind of blocking (traffic jam, queuing) as it is approved in the Model α. Model β is based on data from Model α (route length and transportation time).

In this model, there was chosen production line #5 with 6 stations ("L5P1", "L5P2", etc.) that are supplied by 6 differerent products (product "A" to "L5P1", "B" to "L5P2", etc.).
This model gives information as improved **MR cycle time**, **volumes** of boxes **transported** in each cycle, the required number of **wagons** and establishes the **safety level**. It also parameterizes the **kanban system**. All these parameters and configuration influence each other.
In the actual logistic system in the factory, there are transported special boxes with parts inside. In the model, these boxes are represented by *transportation unit*. This transportation unit has also the same meaning as kanban unit for describing the kanban system.

In the previous model, the loading time is an exact value of 45 seconds. In this model, loading times are based on the number of transported items:

```
loading time in warehouse = number of items x 10 sec (constant value)
```

Establishing kanban unit number in the system

A simplification is used in the simulation model - kanban cards are not used but transportation boxes (transportation unit) are changed one empty box by one full box in the stations and in the supermarket (full by empty). The adjustment of their number is a very important system parameter because a low number can stop the production (lack of material) whereas a high number can be a wasting (excessive storage near production links, extra space, etc.). It is influenced by several factors as:

- production specification
- consumption of material within a time period (average vs. maximum)
- number of parts represented by one kanban card (transportation batch)
- reaction time of supplier (external lead time)
- transportation time between supermarket and station
- quality of parts (percentage of unconformity of input parts)
- desired safety level of storages to covers errors in the production, breakdowns, etc.

Krieg (2005) and Satoh (2008) use the following deterministic calculation to establish kanban units in the system:

$$N = \frac{\mathrm{D}.\,(\mathrm{T}_w + \mathrm{T}_\mathrm{p}\,)\,(1 + \mathrm{a})}{\mathrm{c}}$$

where:

D = consumption of parts by time unit [parts per hour]

T_w = waiting time for kanban batch, transportation times [hours]

T_p = time for processing batch kanban parts, e.g. by assembly operation [hours]

a = safety coefficient

c = number of parts in one kanban transportation unit (capacity)

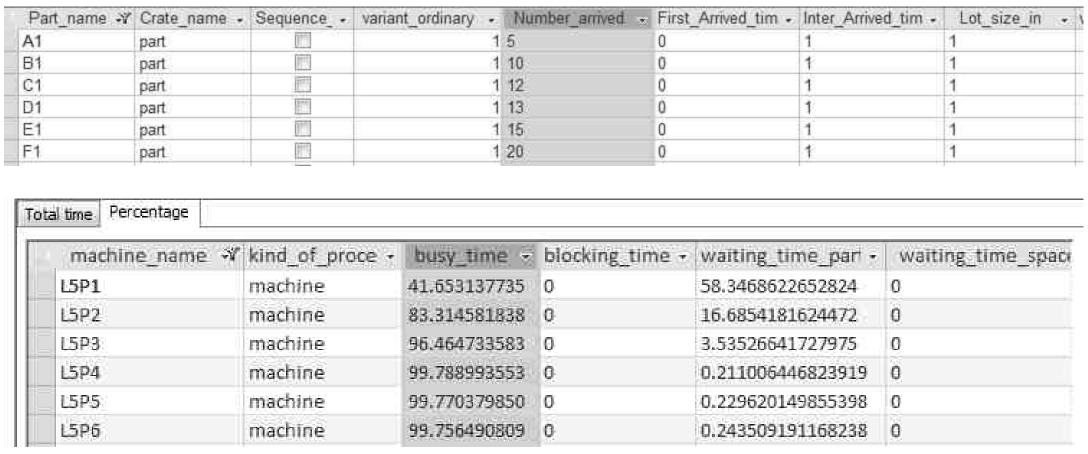
The major problem is to quantify **the safety coefficient** (a) covering many factors as listed above. This value can be from 0.2 up to 9. Due to the difficulty and inaccuracy, the **simulation can be used** to establish the appropriate number of kanban units and to deal with all uncertainties.

One complex simulation model can be split into several experiments to understand more easily **particular results** and to demonstrate **relationships** between all factors, which are: the number of kanban units, storage safety level of items in buffers during milk-run cycle time and different consumption of supplied items.

a) <u>Experiments A: Number of kanban units</u>

This set of experiments shows the influence of kanban units number in the system. In Figure 133 (upper table), there is a set of input data with different number of kanban units in the system for each product (5-10-12, etc).

The aim is to reach 100% working production line in the stations ("L5P1") The lower table of Figure 133 shows the percentage of use of each stations. Results show that values 13 kanban units, enables ~100% of working time. This experiment is for a cyle time of 1200 seconds, with the same consuming speed of 1 box per 200 seconds.



Part_name	Crate_name	Sequence_	variant_ordinary	Number_arrived	First_Arrived_tim	Inter_Arrived_tim	Lot_size_in
A1	part	☐	1	5	0	1	1
B1	part	☐	1	10	0	1	1
C1	part	☐	1	12	0	1	1
D1	part	☐	1	13	0	1	1
E1	part	☐	1	15	0	1	1
F1	part	☐	1	20	0	1	1

Total time | Percentage

machine_name	kind_of_proce	busy_time	blocking_time	waiting_time_pari	waiting_time_spac
L5P1	machine	41.653137735	0	58.3468622652824	0
L5P2	machine	83.314581838	0	16.6854181624472	0
L5P3	machine	96.464733583	0	3.53526641727975	0
L5P4	machine	99.788993553	0	0.211006446823919	0
L5P5	machine	99.770379850	0	0.229620149855398	0
L5P6	machine	99.756490809	0	0.243509191168238	0

Figure 133 - Model β: number of kanban boxes

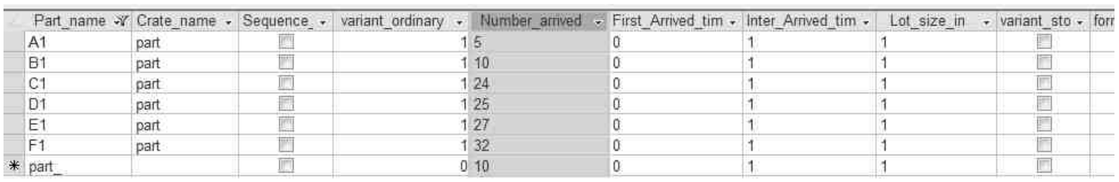
b) <u>Experiments B: Safety level</u>

Safety level means the minimum number of items in buffers that must be there for safety reasons. This number should be enough to cover of at least 2 unrealized MR cycles to avoid that the production stops because of human mistakes, breakdowns, low quality of delivered parts or other unpredictable reasons. In the system, where the MR cycle length is of 1200 seconds and has an average consumption time of 1 box in 200 seconds, the safety level value is 12 units (2*1200/200).

This value (12) is added to the value established in Experiment A (13), so the number of kanban units in the system is established to 25 (12+13). In Figure 134 (upper table), there are different kanban units in the system. For the parts "B1" and "D1", there are 10 kanban units and an established value of 25. The plot graphs in Figure 134 (lower image) display changes in the buffer's occupancy during the simulation: the lower curve is for the buffer without safety level and the upper curve is for the buffer with safety level 12. The number of kanban units does not achieve values under this safety level.

In the beginning of the plot (until time = 2400 seconds), there is a warm-up period of simulation (filling of empty buffers).

The Experiment D, describes the process of establishing safety level in the system influenced by stochastic processes.



	Part_name	Crate_name	Sequence_	variant_ordinary	Number_arrived	First_Arrived_tim	Inter_Arrived_tim	Lot_size_in	variant_sto	forr
	A1	part	☐	1	5	0	1	1	☐	
	B1	part	☐	1	10	0	1	1	☐	
	C1	part	☐	1	24	0	1	1	☐	
	D1	part	☐	1	25	0	1	1	☐	
	E1	part	☐	1	27	0	1	1	☐	
	F1	part	☐	1	32	0	1	1	☐	
*	part_		☐	0	10	0	1	1	☐	

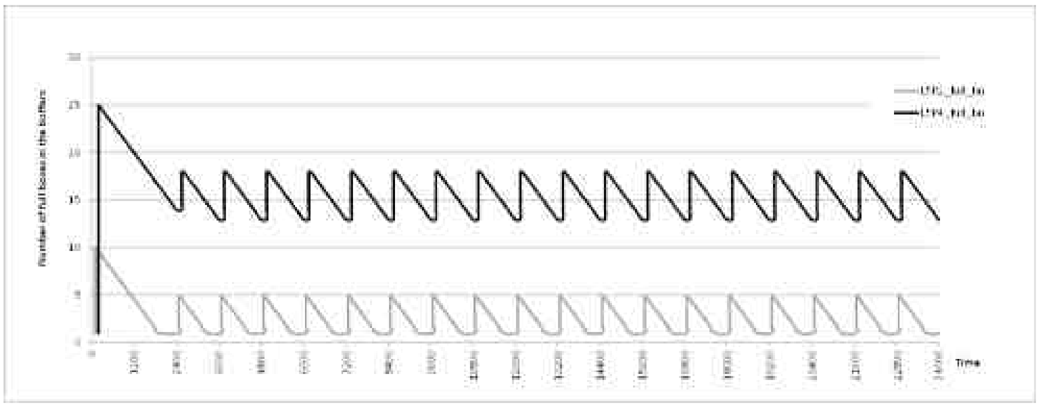


Figure 134 - Model β: safety levels in buffers

c) Experiments C: buffer size based on a different MR cycle time

In this experiment, it is analysed the consequences of increasing the cycle time of MR. The experiment involves the same speed consumption of stations (1 box in 200 seconds). The minimum value for MR was established in the Model α – 1200 seconds (20 minutes). The absolute minimum MR cycle time is 13 minutes. This time is needed just for moving by the MR with the longest transportation route.

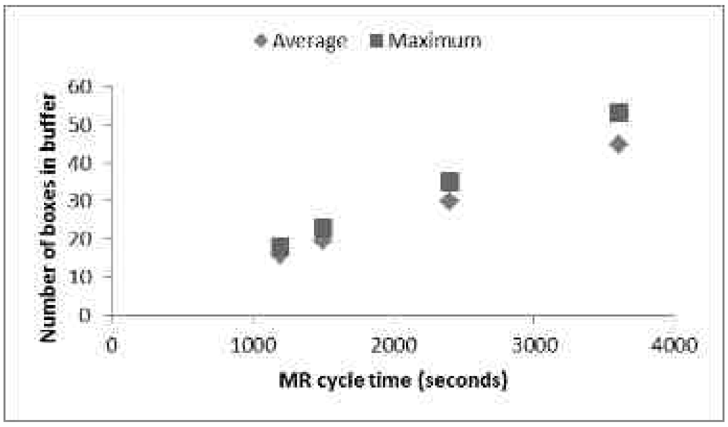
There is a **linear correlation** (Figure 135) that describes the buffer size – longer period means requirement for larger buffers, whereas shorter period can cause an inefficient use of MRs (higher percentage of time of useless movements, or even running empty).

The maximum number of boxes in MR defines the load capacity and so the number of wagons. One wagon can carry around 20 boxes. For a MR cycle of 3600 seconds, it may be necessary to use 162/20 = 8 wagons. More than 3 wagons compromise the MR manoeuvrability.

d) Experiment D: Different consumption of stations

This experiment evaluates the impact of different levels of time consumption of boxes in the stations.

It is relevant to analyse the speed consumption of the delivered items, which is influenced by external customer demands. These demands can be very hard to predict. Therefore, several scenarios are tested (different speed consumptions) to assure if the MR configuration is correct. The MR cycle time is of 1200 seconds. In Figure 136, the set of tables shows input data and experiment results. Different operation times represent different consumption of delivered items. The definition is based on the stochastic function *"Normal(200,40)"*, which means a normal distribution function with a mean = 200 seconds and standard deviation = 40 seconds. Different numbers of kanban units are also defined in the system, based on the average consumption.



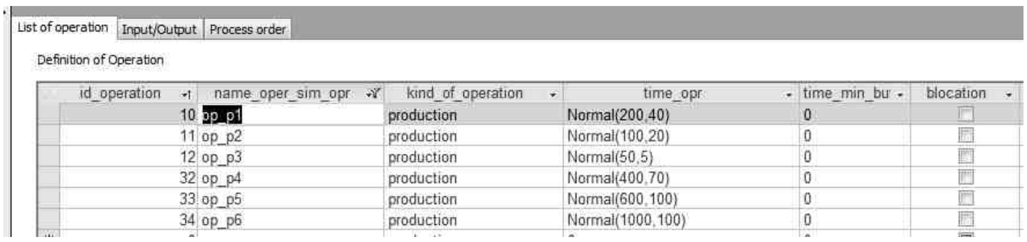
Number of inputing "Parts D"	25	28	36	48
Safety level	12	15	24	36
Number of kanban units in the buffer ("Parts D")	25	31	48	72
MR cycle time [seconds]	1200	1500	2400	3600
Average (after stabilisation 2 MR cycles)	15.68	19.64	29.76	44.8
Maximum number of boxes in buffer	18	23	35	53
Maximum number of boxes in MR	35	54	96	162

Figure 135 - Model β: Plot of the buffer size based on the MR cycle times

Results of the use (Figure 136, third table) display that the number of items ("D1","E1" and "F1") should be increased because there is a lack of items and production line facilities (L5P4, L5P5 and L5P6). Also, they do not work on 100 %. After adding more units into the system, all stations of production line work on 100%.

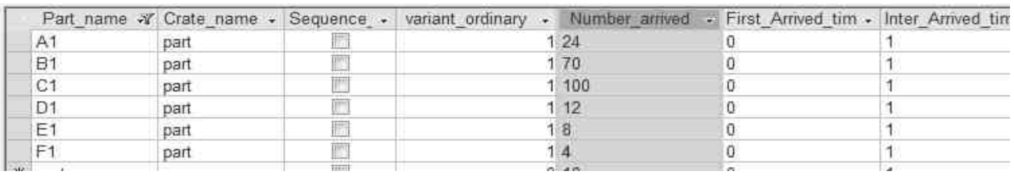
Basically, safety levels are established as the number of boxes consumed within two MR cycles (see Experiment B). If a time is described stochastically, the mean of the distribution is taken and tested. For example, considering Normal (100,20), the safety level can be 2*1200/100 = 24. This value is tested by simulation experiment and results are shown (Figure 136).

In the plot of full boxes in the buffer ("L5P5") (see Figure 137), it happens to be lower than the safety level (= 2) so there is a risk to achieve zero and a consequent breakdown. Therefore, the number of kanban units must be increased. In the second plot, the safety level is 24 and the minimum number during simulation is 20. This can be a point for discussion, if it is necessary to add more kanban units into the logistic system or not, and have a system in a little risky behaviour.

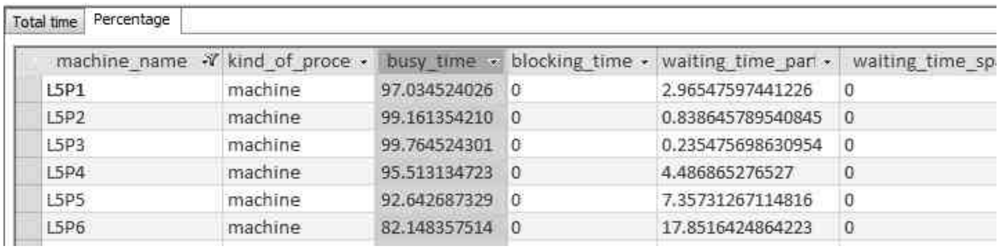

List of operation | Input/Output | Process order

Definition of Operation

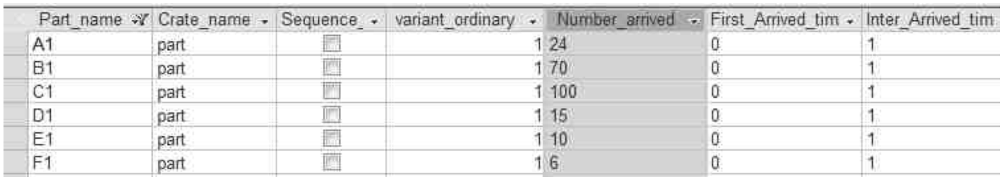
id_operation	name_oper_sim_opr	kind_of_operation	time_opr	time_min_bu	blocation
10	op_p1	production	Normal(200,40)	0	☐
11	op_p2	production	Normal(100,20)	0	☐
12	op_p3	production	Normal(50,5)	0	☐
32	op_p4	production	Normal(400,70)	0	☐
33	op_p5	production	Normal(600,100)	0	☐
34	op_p6	production	Normal(1000,100)	0	☐



Part_name	Crate_name	Sequence_	variant_ordinary	Number_arrived	First_Arrived_tim	Inter_Arrived_tim
A1	part	☐	1	24	0	1
B1	part	☐	1	70	0	1
C1	part	☐	1	100	0	1
D1	part	☐	1	12	0	1
E1	part	☐	1	8	0	1
F1	part	☐	1	4	0	1

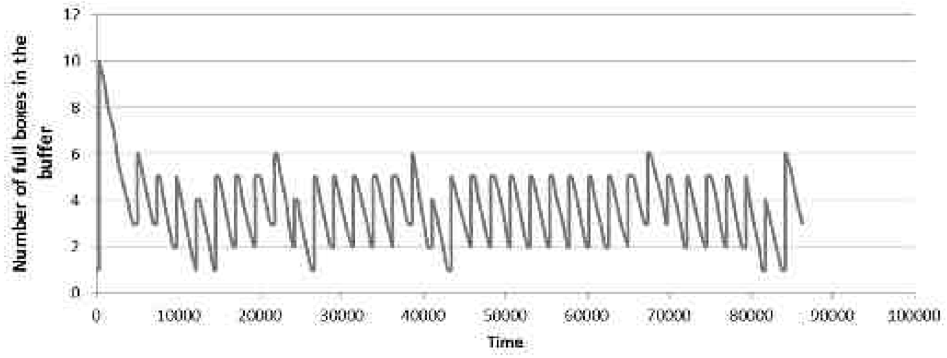

Total time | Percentage

machine_name	kind_of_proce	busy_time	blocking_time	waiting_time_pari	waiting_time_sp
L5P1	machine	97.034524026	0	2.96547597441226	0
L5P2	machine	99.161354210	0	0.838645789540845	0
L5P3	machine	99.764524301	0	0.235475698630954	0
L5P4	machine	95.513134723	0	4.486865276527	0
L5P5	machine	92.642687329	0	7.35731267114816	0
L5P6	machine	82.148357514	0	17.8516424864223	0



Part_name	Crate_name	Sequence_	variant_ordinary	Number_arrived	First_Arrived_tim	Inter_Arrived_tim
A1	part	☐	1	24	0	1
B1	part	☐	1	70	0	1
C1	part	☐	1	100	0	1
D1	part	☐	1	15	0	1
E1	part	☐	1	10	0	1
F1	part	☐	1	6	0	1

Figure 136 - Model β: Different consuming speed



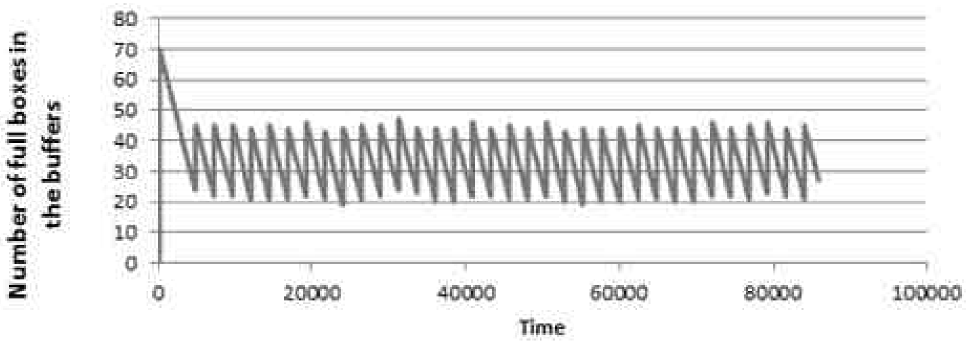
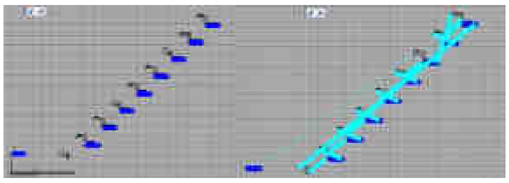
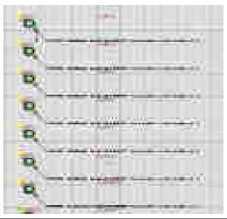
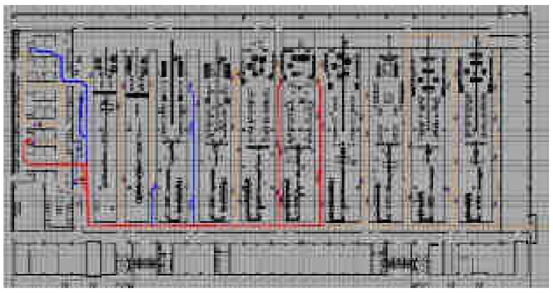
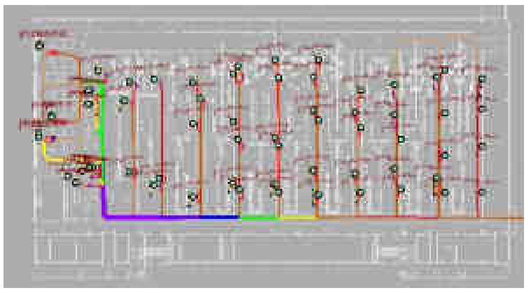
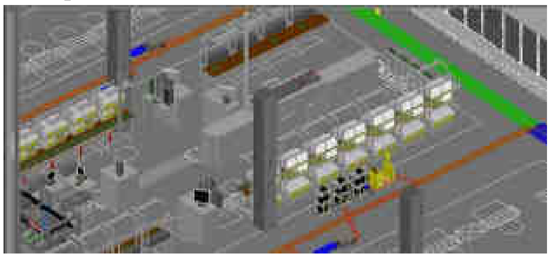


Figure 137 - Model β: Safety level based on stochastic influences

Table 26 – Level of details in IDS

	Level of graphical visualization in IDS	
	CAD layout drawing	**WITNESS simulation model**
I.	Generated block layout Generated material flows 	Generated model 
II.	2D layout Resources placement Transportation flows based on transportation network 	Position of simulation objects based on CAD layout (green rectangulars representing stations, on the background is CAD layout) 
III.	3D layout Transportation flows 	Run time, Virtual reality – 3D animation built by Mantra4D application (not done)

5.2.7 Overall results and comparison

This project described the design of the MR supplying system of internal logistics, solved by IDS. The solution was split into 2 models. The first model describes system with all MRs by simple approach. Results are designed routes and transportation times. The second model describes in details the behaviour of one MR. This behaviour and the MR configuration are divided into several experiments based on different parameters: the number of kanban units, milk-run cycle times, holding the safety level in buffers and the different consumption of supplied items.

Based on these simulation models, the MR system can be configured and it is flexible to be used in different scenarios.

This project shows IDS abilities as an integration of SIM and CAD, rapid design, time-efficient and different levels of details in designing (general and detailed, Model α and Model β).

As mentioned before, this example was previously solved with the simulation tool Arena. The final results data of both projects are similar. In summary, manually made model needs **4 weeks** (without time for data collection and analysis, and creating 3D animation). IDS needs around **3 days** (see Table 34).

Images from Arena projects are in Appendix (Section A.5.2).

It is possible to design logistic or production system in several graphic levels based on details, as shown in the images in Table 26.

5.3 Magna

This project is focused on the internal logistic in automotive industry. The company "Magna Exteriors & Interiors" (until 2009 under the name Cadence Innovation) is a producer and designer of plastic parts (www.cadenceinnovation.cz, 2008)(www.magna.com, 2011).

It was set up in 1946 and started by producing plastic parts for kitchen and garden. Since 1982, it has produced plastic parts for the automotive industry, such as: painted bumpers (33%, around 4300 per day), control desks (38%, 3000 per day), door fillers (22%, 6200 per day) and grid of cooler (7%) in 2008.

It is a successful company with a current turnover of around 300 million €, and it has grown as one of the largest companies in the Liberec district (supplier for ŠKODA, VW[1] and OPEL) with 2500 employees. It also spread to other Czech towns – Nymburk (2007 – supplier of TPCA[2] and AUDI) and Liban (2007) - and to Hungary (2007, Esztergom, SUZUKI). It is now preparing the expansion to Russia (2011).

5.3.1 Project overview and description of processes

This project is focused on the production of bumpers and internal logistics linked with it. A similar topic was solved as a project in 2007 and 2008 (Jareš, 2008)(Vik & Jareš, 2008)(Vik, Dias, Pereira, & Oliveira, 2010d).

Products description

The factory in Liberec City produces parts for 5 Škoda car models, 4 part types for each of them (front and rear bumpers, central strips, front grids). This project involves only "big parts" (bumpers), while "small parts" (strips, grids) are omitted because their production is independent on the bumpers on which this project is concentred. There are 21 colours for car models. The combination of colour and car model (part type) is named by the term **"colour-type"** (**CT**) that is the name used for **painted bumpers**. Not every car model is painted in a full colour set but

[1] Volkswagen

[2] Toyota – Peugeot – Citroen Automobile

only in a defined colour set, shown in Figure 90. In some cases, it can appear non-standard colours for special customers (police, taxis, companies, etc.). These colours are also omitted in these models.

Production and logistic processes overview

The complete production processes are in Figure 139, and the factory layout in Figure 148.

Injection moulding machines produce non-coloured bumpers (**batch production**). According to the amounts of items in the "Warehouse of coloured parts", non-coloured bumpers are sent to the "Paint shop". This process is controlled by kanban pull system of orders that cares to hold the established level of products in the Warehouse (so called "safety level").

After the painting operations, CTs are transported by a conveyor to the check quality workstations ("*Check WS*"), where quality is checked and small visual defects are rectified by brushing and polishing. After that, CTs are hang on another conveyor, transported near to the "Warehouse" and stored there in special containers (crates).

Based on customer orders, CTs are removed from the Warehouse, assembled and dispatched to the customers. This part of logistics system is based on the JIT principle ("Just-In-Time").

Follows the detailed description of each process:

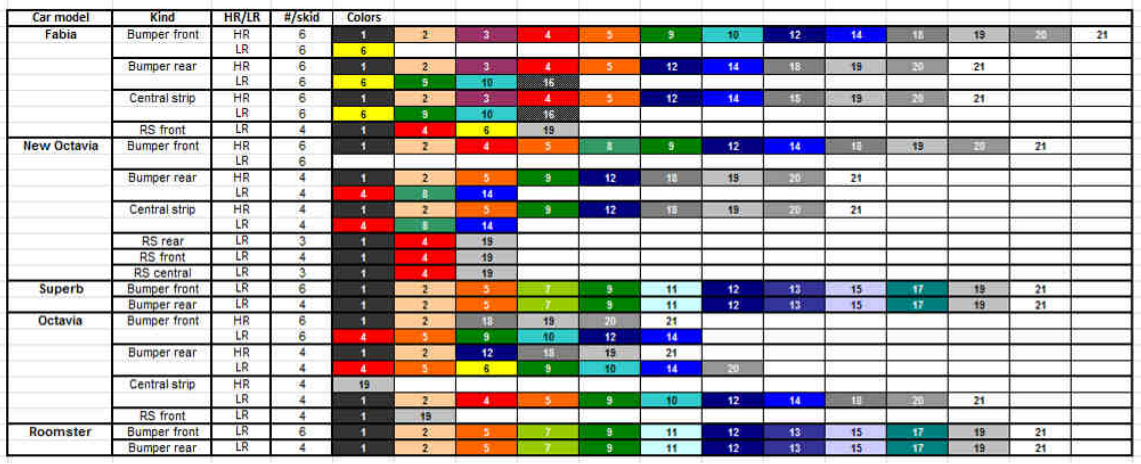
a) Paint shop

Paint shop consists of two painting lines divided by colours and type on the "High-runners"(HR) and "Low-runners"(LR), based on the production volumes (large-size lots and small-size lots). HRs (e.g. silver FABIA bumpers) are produced every day in a larger amount. This division of CTs is due to changing colours in the painting robots, cleaning and making their setup. LR production (e.g. racing car models – RS) consists of small series and colours that are changed very often, despite the fact that painting robots use the same colours for HR.

Transport between "Storage of non-coloured parts" through the paint shop is managed by skids hung on the conveyor. Skid is a special cage that has attachments for each type of bumpers and its capacity is derived from the bumper's size (from 3 to 8 bumpers). Conveyor functions are the **transport** of skids and also a buffer to **balance** time differences between tact of paint shop and Check Quality WS.

Outgoing skid sequences are established with bumpers in the rate 4(HR):1(HR) corresponding to the required amount of CTs.

The "small parts" (as mouldings, grids etc.) are also painted, and these skids are inserted into the outputting stream randomly without any strict rules.



Car model	Kind	HR/LR	#/skid	Colors												
Fabia	Bumper front	HR	6	1	2	3	4	5	9	10	12	14	18	19	20	21
		LR	6	6												
	Bumper rear	HR	6	1	2	3	4	5	12	14	18	19	20	21		
		LR	6	6	9	10	16									
	Central strip	HR	6	1	2	3	4	5	12	14	15	19	20	21		
		LR	6	6	9	10	16									
	RS front	LR	4	1	4	6	19									
New Octavia	Bumper front	HR	6	1	2	4	5	8	9	12	14	18	19	20	21	
		LR	6													
	Bumper rear	HR	4	1	2	5	9	12	18	19	20	21				
		LR	4	4	8	14										
	Central strip	HR	4	1	2	5	9	12	18	19	20	21				
		LR	4	4	8	14										
	RS rear	LR	3	1	4	19										
	RS front	LR	4	1	4	19										
	RS central	LR	3	1	4	19										
Superb	Bumper front	LR	6	1	2	5	7	9	11	12	13	15	17	19	21	
	Bumper rear	LR	4	1	2	5	7	9	11	12	13	15	17	19	21	
Octavia	Bumper front	HR	6	1	2	18	19	20	21							
		LR	6	4	5	9	10	12	14							
	Bumper rear	HR	4	1	2	12	18	19	21							
		LR	4	4	5	6	9	10	14	20						
	Central strip	HR	4	19												
		LR	4	1	2	4	5	9	10	12	14	18	20	21		
	RS front	LR	4	1	19											
Roomster	Bumper front	LR	6	1	2	5	7	9	11	12	13	15	17	19	21	
	Bumper rear	LR	4	1	2	5	7	9	11	12	13	15	17	19	21	

Figure 138 - Type-colour table

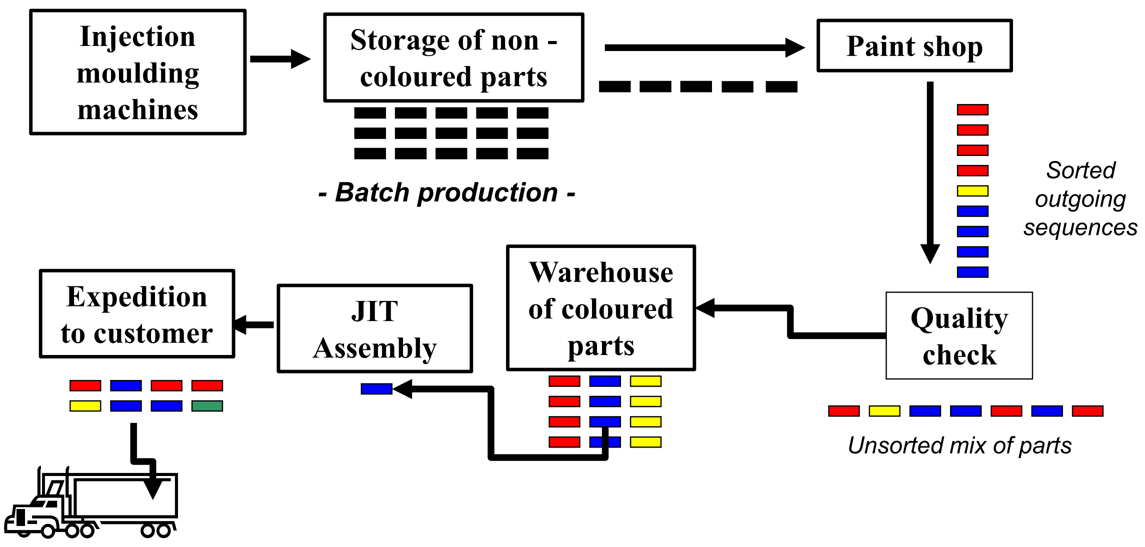

Injection moulding machines
Storage of non - coloured parts
- Batch production -
Paint shop
Sorted outgoing sequences
Quality check
Unsorted mix of parts
Warehouse of coloured parts
JIT Assembly
Expedition to customer

Figure 139 – Schema of production

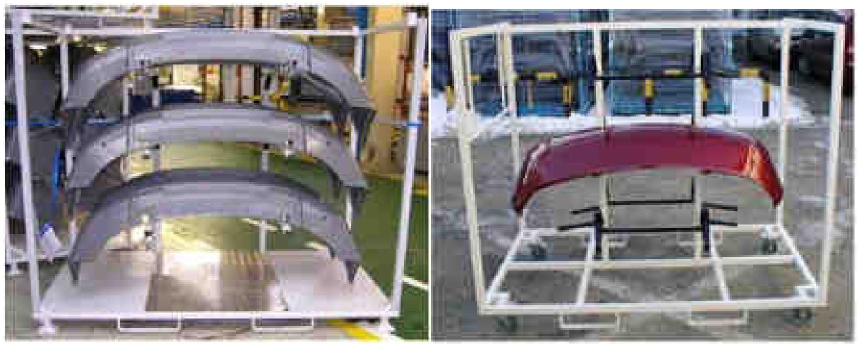


Figure 140 – Transportation units for coloured bumpers

b) Check Quality Workstations

After the painting processes, bumpers are hung on another conveyor system and go to the quality checkpoint ("Check workstations", Check WS). Each CT is taken from the conveyor and its quality is checked. Some bumpers need to be polished by a special brush machine or locally repainted. Products with low quality are recycled.

The operation time depends on the quality, size of area for polishing and the product shape.

It happens that the sorted product sequences from the paint shop are mixed – some bumpers are completely removed from the system, some are delayed to the next sequence, some of them are returned back to paint shop.

c) <u>Warehouse</u>

CT are taken down from the conveyor and inserted into the transport crates (cages or containers) and sorted by type and colour (see examples in Figure 140). Each product type has a different crate based on the size and shape.

Full-loaded crates are transported by forklifts and stored in the Warehouse. Drivers of forklifts also prepare empty crates for incoming CT sequences. This information is shown on the electronic board in order to have enough time to prepare them.

Crates, that are partially loaded, must wait for the next same CT sequence in the floating temporary storage.

The warehouse has also the function given by the main customer – holding an appropriate amount of each CT for two days of production. It is called **safety level** of storage.

The warehouse must work on the FIFO principle and there is also one special rule: parts must stay there a couple of hours to achieve the flash time of paintings.

The Paint shop scheduling is based on internal orders from the Warehouse. Schedules are prepared manually by paint shop manager decisions. The preparation of sequences is done by a sophisticated forecast, which considers the actual conditions, such as the output quality of each CTs, priorities, avoiding changing colours and setting up robots or other special customer demands.

d) <u>JIT assembly</u>

Based on the JIT orders, bumpers (CTs) and other components are delivered to the assembly workplaces from the Warehouse. It was described in the project introduction.

5.3.2 Project aims

This production and logistic system are complex. They describe the area between the batch production and the JIT assembly. It is very helpful to make analysis and get answers for some project tasks.

The current production per day is around 6000 bumpers. In the future, this volume should be **increased** up to 10 000 produced in the same paint shop facility and the number of CTs will be higher than the current one (from 96 up to 220). Reasons are clear, production will spread with the new model of car, facelifts of the current models and new colours as well as requirements for spare parts. These mentioned facts cause the grouth of product variants. The need is to produce simultaneously old models as well as new models because of the spare parts. It is necessary to test it and recognise necessary system changes.

The project in **IDS** should be helpful to acquire the following information:

- Necessary number of brushing operators (Check Quality workstations), number of operators for the take-down operation and forklift drivers, use of those resources
- Size of the buffer and storages (mainly temporary floating buffer and the Warehouse)
- Condition of holding a safety level in the storages
- Required number of containers in the system
- Give helpful information in paint shop scheduling

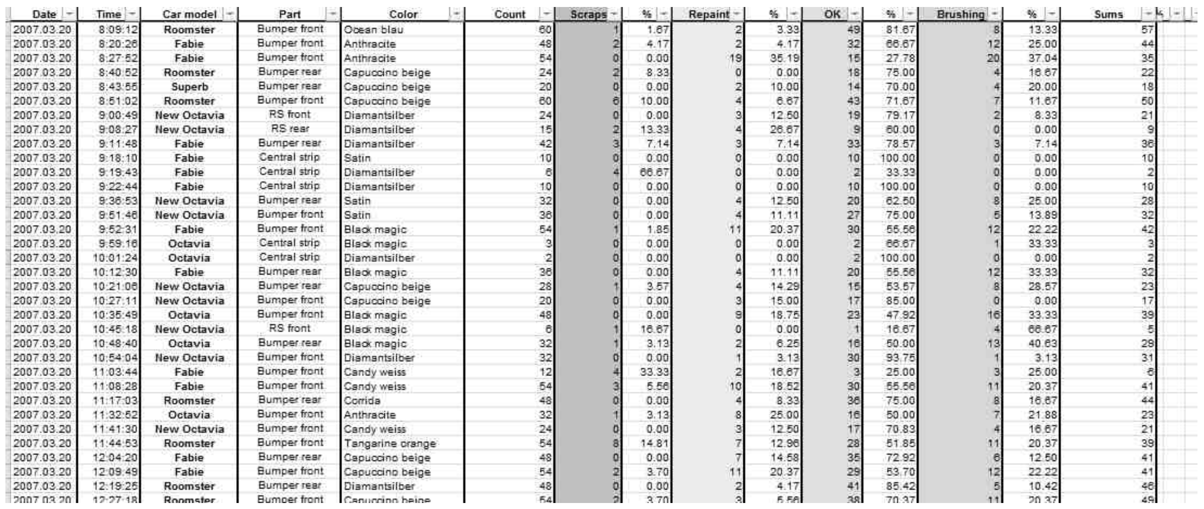
For this purpose, a simulation model is built and a set of experiments is run, required input data are processed and analysed in the following section. This model can be used for testing different paint shop schedules.

5.3.3 Analysis and data processing

This section contains data analysis required for the simulation model. Schematic (conceptual) model is shown in Figure 145. It shows relationships in the area. The processed data are then written into the project DB.

a) Paint shop data processing

Data about product arrivals are taken from production data set, provided by company´s ERP system. That data is recorded in the check workstations where every labour must save information about the product into the system with the help of barcodes. These data can be used as schedules of paint shop. **Production data** consists of time (date), car model, CT name, colour, number of parts, and number of scraps (repaint, OK and part for brushing) (see Figure 141).



Date	Time	Car model	Part	Color	Count	Scraps	%	Repaint	%	OK	%	Brushing	%	Sums
2007.03.20	8:09:12	Roomster	Bumper front	Ocean blau	60	1	1.67	2	3.33	49	81.67	8	13.33	57
2007.03.20	8:20:26	Fabie	Bumper front	Anthracite	48	2	4.17	2	4.17	32	66.67	12	25.00	44
2007.03.20	8:27:52	Fabie	Bumper front	Anthracite	54	0	0.00	19	35.19	15	27.78	20	37.04	35
2007.03.20	8:40:52	Roomster	Bumper rear	Capuccino beige	24	2	8.33	0	0.00	18	75.00	4	16.67	22
2007.03.20	8:43:56	Superb	Bumper rear	Capuccino beige	20	0	0.00	2	10.00	14	70.00	4	20.00	18
2007.03.20	8:51:02	Roomster	Bumper front	Capuccino beige	60	6	10.00	4	6.67	43	71.67	7	11.67	50
2007.03.20	9:00:49	New Octavia	RS front	Diamantsilber	24	0	0.00	3	12.50	19	79.17	2	8.33	21
2007.03.20	9:08:27	New Octavia	RS rear	Diamantsilber	15	2	13.33	4	26.67	9	60.00	0	0.00	9
2007.03.20	9:11:48	Fabie	Bumper rear	Diamantsilber	42	3	7.14	3	7.14	33	78.57	3	7.14	36
2007.03.20	9:18:10	Fabie	Central strip	Satin	10	0	0.00	0	0.00	10	100.00	0	0.00	10
2007.03.20	9:19:43	Fabie	Central strip	Diamantsilber	6	4	66.67	0	0.00	2	33.33	0	0.00	2
2007.03.20	9:22:44	Fabie	Central strip	Diamantsilber	10	0	0.00	0	0.00	10	100.00	0	0.00	10
2007.03.20	9:36:53	New Octavia	Bumper rear	Satin	32	0	0.00	4	12.50	20	62.50	8	25.00	28
2007.03.20	9:51:46	New Octavia	Bumper front	Satin	36	0	0.00	4	11.11	27	75.00	5	13.89	32
2007.03.20	9:52:31	Fabie	Bumper front	Black magic	54	1	1.85	11	20.37	30	55.56	12	22.22	42
2007.03.20	9:59:16	Octavia	Central strip	Black magic	3	0	0.00	0	0.00	2	66.67	1	33.33	3
2007.03.20	10:01:24	Octavia	Central strip	Diamantsilber	2	0	0.00	0	0.00	2	100.00	0	0.00	2
2007.03.20	10:12:30	Fabie	Bumper rear	Black magic	36	0	0.00	4	11.11	20	55.56	12	33.33	32
2007.03.20	10:21:06	New Octavia	Bumper rear	Capuccino beige	28	1	3.57	4	14.29	15	53.57	8	28.57	23
2007.03.20	10:27:11	New Octavia	Bumper front	Capuccino beige	20	0	0.00	3	15.00	17	85.00	0	0.00	17
2007.03.20	10:35:49	Octavia	Bumper front	Black magic	48	0	0.00	9	18.75	23	47.92	16	33.33	39
2007.03.20	10:45:18	New Octavia	RS front	Black magic	6	1	16.67	0	0.00	1	16.67	4	66.67	5
2007.03.20	10:48:40	Octavia	Bumper rear	Black magic	32	1	3.13	2	6.25	16	50.00	13	40.63	29
2007.03.20	10:54:04	New Octavia	Bumper front	Diamantsilber	32	0	0.00	1	3.13	30	93.75	1	3.13	31
2007.03.20	11:03:44	Fabie	Bumper front	Candy weiss	12	4	33.33	2	16.67	3	25.00	3	25.00	6
2007.03.20	11:08:28	Fabie	Bumper front	Candy weiss	54	3	5.56	10	18.52	30	55.56	11	20.37	41
2007.03.20	11:17:03	Roomster	Bumper rear	Corrida	48	0	0.00	4	8.33	36	75.00	8	16.67	44
2007.03.20	11:32:52	Octavia	Bumper front	Anthracite	32	1	3.13	8	25.00	16	50.00	7	21.88	23
2007.03.20	11:41:30	New Octavia	Bumper front	Candy weiss	24	0	0.00	3	12.50	17	70.83	4	16.67	21
2007.03.20	11:44:53	Roomster	Bumper front	Tangarine orange	54	8	14.81	7	12.96	28	51.85	11	20.37	39
2007.03.20	12:04:20	Fabie	Bumper rear	Capuccino beige	48	0	0.00	7	14.58	35	72.92	6	12.50	41
2007.03.20	12:09:49	Fabie	Bumper front	Capuccino beige	54	2	3.70	11	20.37	29	53.70	12	22.22	41
2007.03.20	12:19:25	Roomster	Bumper rear	Diamantsilber	48	0	0.00	2	4.17	41	85.42	5	10.42	46
2007.03.20	12:27:18	Roomster	Bumper front	Capuccino beige	54	2	3.70	3	5.56	38	70.37	11	20.37	49

Figure 141 - Paint shop data

b) P-Q analysis

Production data is summarised and the P-Q analysis is processed. Figure 142 shows production volumes for each CT during the chosen week.

Product portfolio varies, being the current 103 CTs. It is not necessary to simulate all of them because of the same properties and behaviour. For the purpose of the simulation model, 10 CT (2 HR, 8 LR) were chosen and they are pointed by arrows, as shown in Figure 142.

The arrival schedules of the chosen HRs and LRs have the same percentage appearance and quantity as in the reality, so the total amount of inputting CT is also the same. The final statistics of facilities corresponds to the real system.

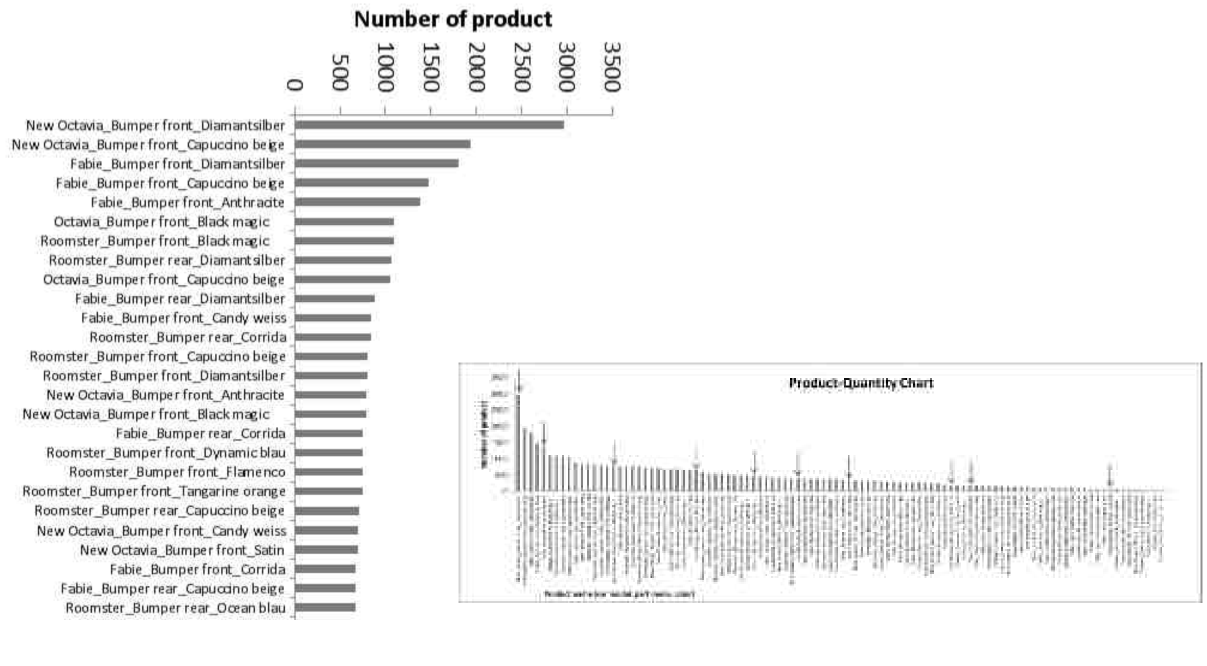


Figure 142 – P-Q analysis

c) Output quality

From the set data (Figure 141), it is also valued the average percentage of quality (scraps, repainting, etc.) - see results in Table 27. Based on that, a **profile** was created in a simulation model and the quality level used the attribute *"vol"*.

Table 27 – Part quality analysis

Quality level	Perfect products	Brushing	Local repainting	Complete repainting	Waster
%	60%	25%	5%	5%	5%
vol	1	2	3	4	5

d) Operation times of Check Workplace

These values were measured by a stopwatch and an analysis of it was made. All of them have *normal distribution* (Normal(*Mean*, *Standard deviation*)).

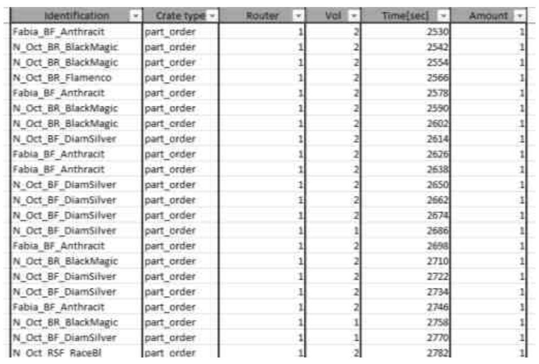
Operation	Mean [seconds]	Standard deviation [seconds]
Checking(1)	40	10
Polishing and brushing(2)	160	40
Checking (repaint, waster)(3,4,5)	20	3
Hang on /Take down of bumpers	10	5
Transportation and storing in the Warehouse	120	60
Assembly operation	300	50
Assembly (packing for resellers)	45	7
Painting	10800 (3 hours)	500

e) JIT orders, analysis and data preparation

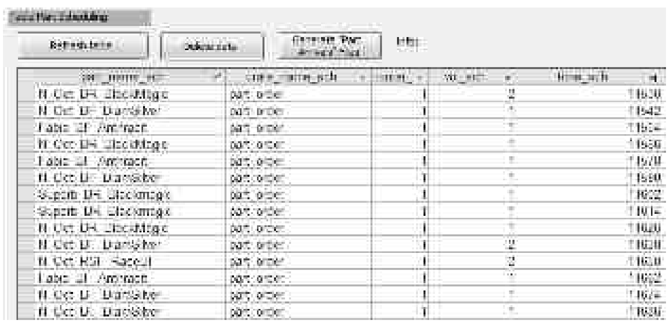
JIT customers' orders come in a very random order. For feeding IDS system, VBA macro in MS Excel generates randomly JIT demands by the following rules (see Figure 143):

- The total percentage of CTs is based on volumes outgoing from the Paint shop,
- 25% of the demands are JIT assembly for direct factory customers, 75% of the demands are for resellers (determinated by *vol* attribute)
- together 5500 parts per one day,
- JIT orders are generated in simplified time patterns: every 12 seconds, one JIT order

In case of the JIT order from resellers, these CTs only pack all the other operation that is done in the workshop of car services or these parts are sold as spare bumpers.



Identification	Crate type	Router	Vol	Time[sec]	Amount
Fabia_BF_Anthracit	part_order	1	2	2530	1
N_Oct_BR_BlackMagic	part_order	1	2	2542	1
N_Oct_BR_BlackMagic	part_order	1	2	2554	1
N_Oct_BR_Flamenco	part_order	1	2	2566	1
Fabia_BF_Anthracit	part_order	1	2	2578	1
N_Oct_BR_BlackMagic	part_order	1	2	2590	1
N_Oct_BR_BlackMagic	part_order	1	2	2602	1
N_Oct_BF_DiamSilver	part_order	1	2	2614	1
Fabia_BF_Anthracit	part_order	1	2	2626	1
Fabia_BF_Anthracit	part_order	1	2	2638	1
N_Oct_BF_DiamSilver	part_order	1	2	2650	1
N_Oct_BF_DiamSilver	part_order	1	2	2662	1
N_Oct_BF_DiamSilver	part_order	1	2	2674	1
N_Oct_BF_DiamSilver	part_order	1	1	2686	1
Fabia_BF_Anthracit	part_order	1	2	2698	1
N_Oct_BR_BlackMagic	part_order	1	2	2710	1
N_Oct_BF_DiamSilver	part_order	1	2	2722	1
N_Oct_BF_DiamSilver	part_order	1	2	2734	1
Fabia_BF_Anthracit	part_order	1	2	2746	1
N_Oct_BR_BlackMagic	part_order	1	1	2758	1
N_Oct_BF_DiamSilver	part_order	1	1	2770	1
N_Oct_RSF_RaceBl	part_order	1	2	2782	1



part_name_sch	crate_name_sch	router_	vol_sch	time_sch
N_Oct_BR_BlackMagic	part_order	1	2	11530
N_Oct_BF_DiamSilver	part_order	1	1	11542
Fabia_BF_Anthracit	part_order	1	1	11554
N_Oct_BR_BlackMagic	part_order	1	1	11566
Fabia_BF_Anthracit	part_order	1	1	11578
N_Oct_BF_DiamSilver	part_order	1	1	11590
Superb_BR_Blackmagic	part_order	1	1	11602
Superb_BR_Blackmagic	part_order	1	1	11614
N_Oct_BR_BlackMagic	part_order	1	1	11626
N_Oct_BF_DiamSilver	part_order	1	2	11638
N_Oct_RSF_RaceBl	part_order	1	2	11650
Fabia_BF_Anthracit	part_order	1	1	11662
N_Oct_BF_DiamSilver	part_order	1	1	11674
N_Oct_BF_DiamSilver	part_order	1	1	11686

Figure 143 – JIT orders analysis

f) Safety level of resource in the Warehouse

Customers also demand holding safety storage level of bumpers, usually for two days. The consumption for two days, for the chosen CT, is in Figure 144 as well as the number of full loaded containers.

type	colour	2 days consumption	# bumpers in the skid	#containers
New Octavia, bumper rear	blackmagic	400	8	50
	cappucino	260	8	33
	corrida	40	8	5
	flamengo	50	8	7
	sprint	50	8	7
	island	40	8	5
	highland	20	8	3
	stormblau	100	8	13
	dynamic	60	8	8
	anthracit	250	8	32
	satin	120	8	15
	brilliant	410	8	52
	candy	200	8	25
		2000		***255***

Figure 144 – JIT order analysis

5.3.4 Simulation

a) Simulation model

A schema of the simulation model is in Figure 145. Text boxes with blue background are buffers or conveyors, text boxes with yellow background represent real facilities and grey text boxes are logic control (virtual) processes that are not in the real system. Flows represent transportation (conveyor, containers or final products), information flows are customers' orders or signals for paint shop.

As mentioned in the previous sections, some simplification is used. The number of CTs is reduced to 10% with the increased production volumes.

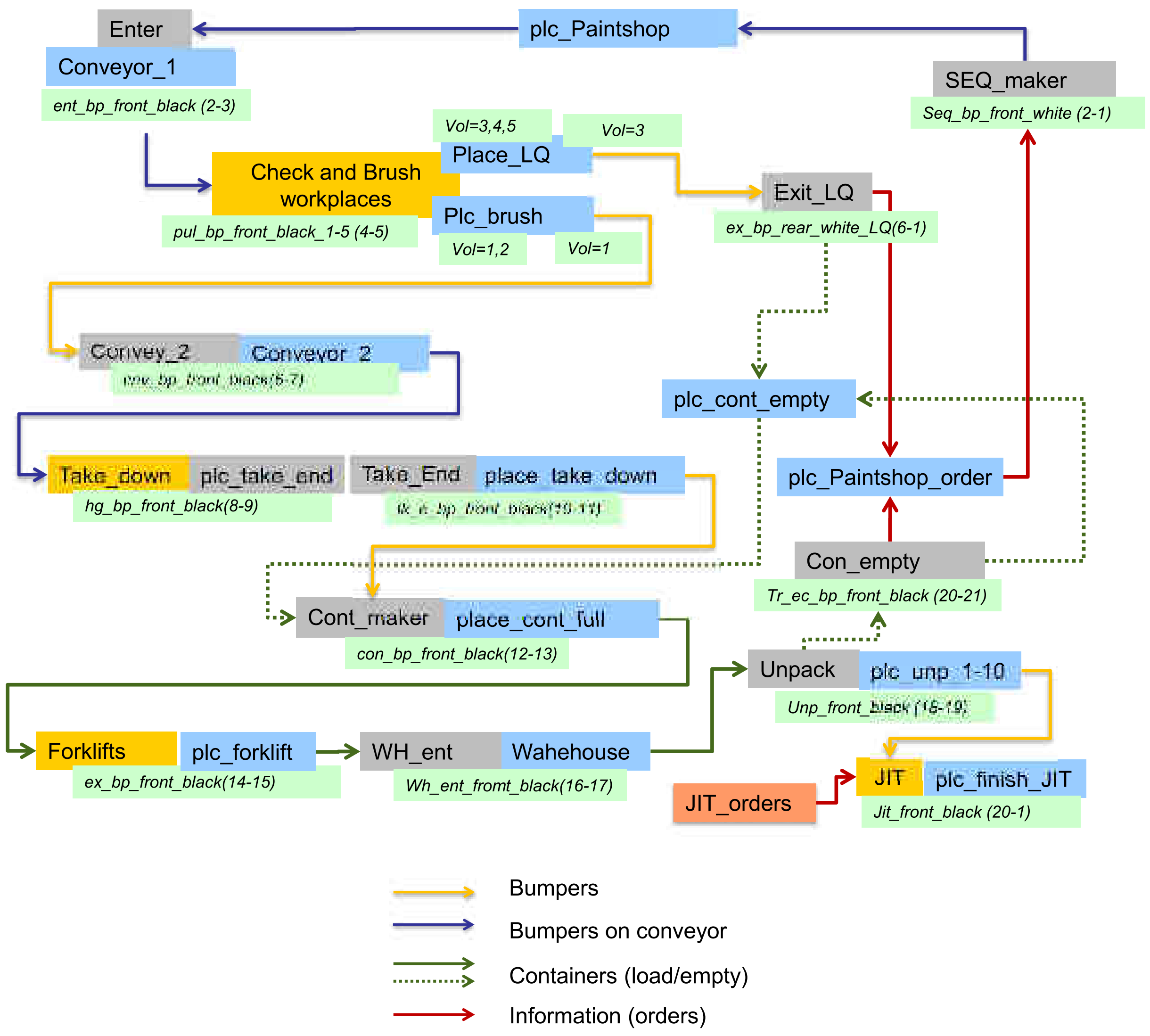


Figure 145 – Schema of the simulation model

Facilities as *Checks Quality*, *Forklifts* and *Take down* are "*quantity*" kind of machine (see Section Section A.4.2). The *quantity* of resources can be easily changed as parameters in the experiments.

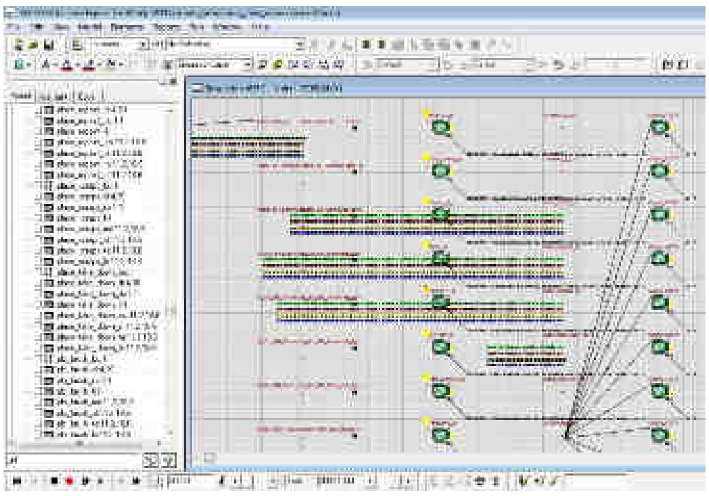


Figure 146 – Magna: Generated simulation model

Brief description of the simulation model

JIT orders are the main inputting entities; all processes are controlled based on them. In a JIT assembly (*"JIT"*), the JIT orders are joined with the corresponding CT. CTs are taken from containers that are unpacked (according to the function described in Section A.4.2.)

Then, empty containers are transported to the storage (*"Con_empty"*) that makes signal for the paint shop (*"plc_Paintshop_order"*) to produce new parts (*"SEQ_maker"*) in sequences of a rate 4:1 (HR:LR). The paint shop (*"plc_Paintshop"*) is represented by a buffer where a part must stay for a defined time, equal to the operation time. Parts going from the paint shop are taken from the conveyor (*"Conveyor_1"*) and checked (*"CheckWP"*). Parts with low quality are sent out (*"place_LQ"*), while the good parts are hung to the conveyor (*"Conveyor_2"*). From there, they are taken down (*"Take_down"*) and stored in the container (*"Cont_maker"*). Full loaded containers are transported by forklifts to the Warehouse (*"WH_ent"*). From the Warehouse, containers are taken and unpacked (*"Unpack"*).

For the transportation by forklifts, it is used the approach described in Section **A.4.5**, for the unpack operation, it is used the approach mentioned in Section A.4.2.

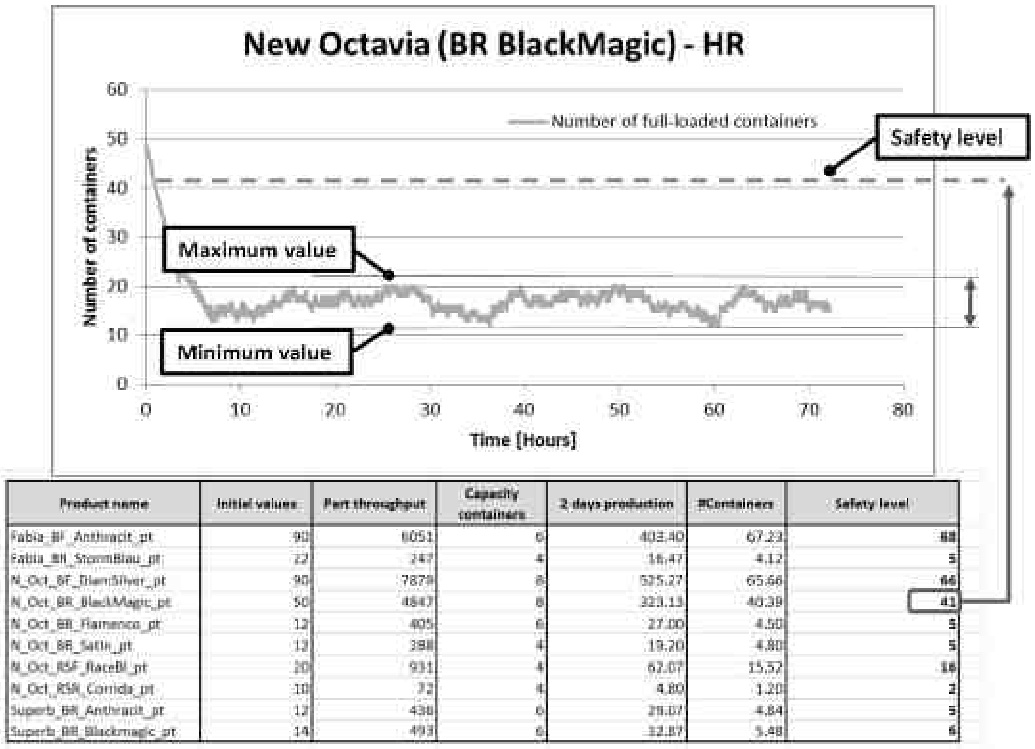
It it necessary to initialise the model before the simulation running by filling buffers with containers and setup the operation for unpacking (Section A.4.3).

The simulation model is automatically generated based on data from DB (see Figure 146).

b) <u>Results of the current state</u>

The number of containers in the system is estimated by basic deterministic calculation. These values must be validated with the simulation results to establish realistic safety levels.

Figure 147 shows a schema of the safety level analysis. Graph results from the initial simulation model shows the number of full-loaded containers for a given CT (New Octavia Black Magic, HR) in the Warehouse during the simulation. After the simulation system stabilised (after 10 hours running), there are between 10-20 full loaded containers in the Warehouse, with a calculated safety level of 41. This value is established by a 2-day production and the recalculated value into full-loaded containers is based on their capacity. Therefore, it must be added at least 30 containers with a given CT to accomplish holding safety level condition. In current state, there are following number of facilities: *"Check WS"* (10), *"Forklifts"* (2), *"Take down"* (2) and *"JIT WS"* (10).



Product name	Initial values	Part throughput	Capacity containers	2 days production	#Containers	Safety level
Fabia_BF_Anthracit_pt	90	6051	6	403.40	67.23	68
Fabia_BR_StormBlau_pt	22	247	4	16.47	4.12	5
N_Oct_BF_DiamSilver_pt	90	7879	8	525.27	65.66	66
N_Oct_BR_BlackMagic_pt	50	4847	8	323.13	40.39	41
N_Oct_BR_Flamenco_pt	12	405	6	27.00	4.50	5
N_Oct_BR_Satin_pt	12	288	4	19.20	4.80	5
N_Oct_RSF_RaceBl_pt	20	931	4	62.07	15.52	16
N_Oct_RSR_Corrida_pt	10	72	4	4.80	1.20	2
Superb_BR_Anthracit_pt	12	436	6	29.07	4.84	5
Superb_BR_Blackmagic_pt	14	493	6	32.87	5.48	6

Figure 147 – Safety level determination

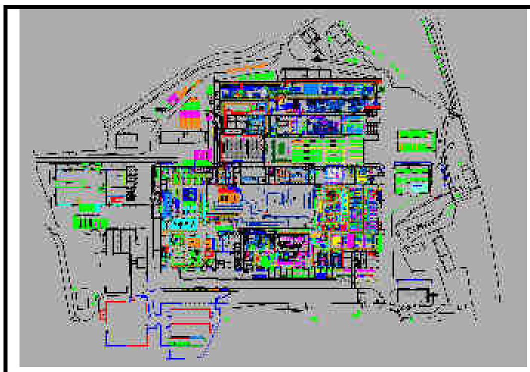
5.3.5 Experiment - increased production

In this experiment, the influences of changes of increased production in the system are studied. There are schedules tested with 10 000 daily production and processed on the current system (paint shop outgoing sequences, operation times, quality).

Simulation models are used the same from the current system model. The chosen results of this experiment are in Table 28. Several different configurations are tested for facilities quantity values to find optimal.

The total throughput during three days is 29985 that corresponds to a 10 000 daily production. The use of "*CheckWS*" is around 90% and forklifts are busy in 64%. The assembly of workstations works on an average of 90%. The total stable number of full-loaded containers in the Warehouse is around 150, without considering safety storages of each CTs.

Results of the experiment display that the established configuration is enough for an increased production.



(1) Mouldering machines
(2) Storage of non-coloured parts
(3) Preparation of black parts
(4) Paint shop
(5) Check Quality Workplaces
(6) Take Down area
(7) Storage of full-loaded containers
(8) JIT assembly workstations
(9) Expedition

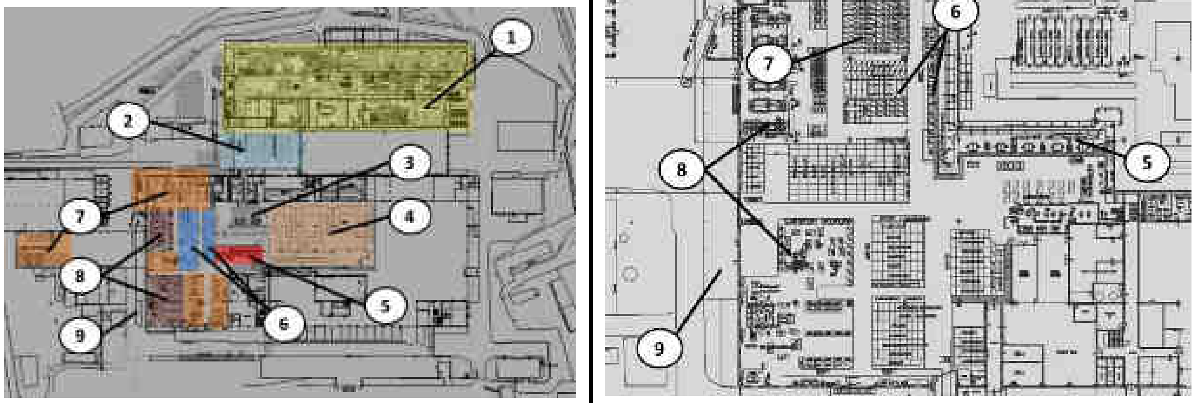


Figure 148 – Magna: layouts

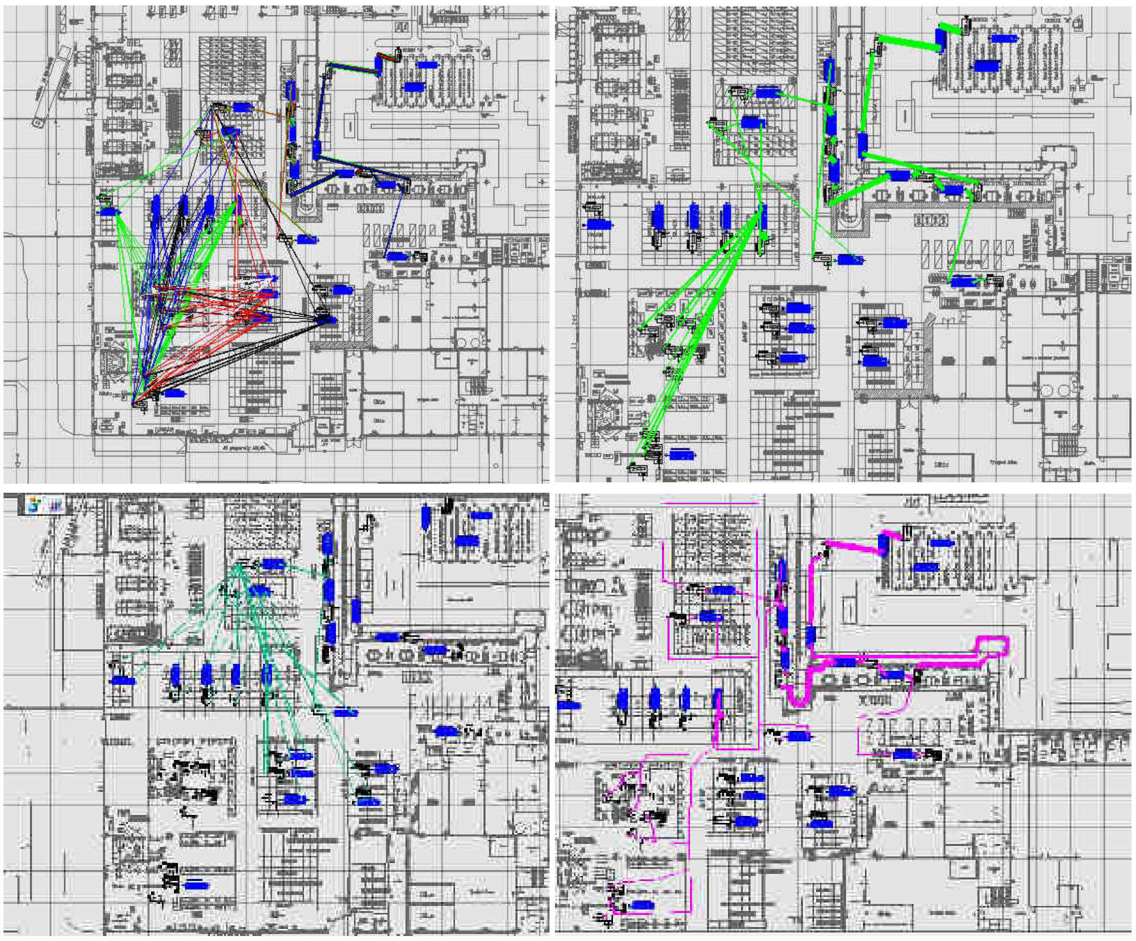


Figure 149 – Magna: layouts

5.3.6 CAD layouts

a) Material flows

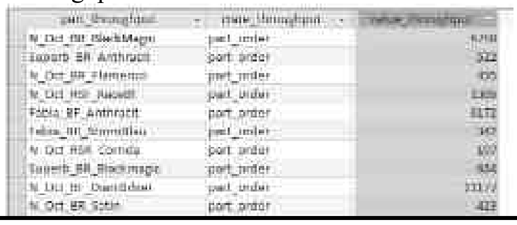
Layouts are very helpful tools for designing production system. In Figure 149, there are several layouts with generated material flows. Resources and buffers are simple schematic drawing blocks; the actual factory layout is on the background (grey colour). The first image displays schematic layout with direct unconstrained material flows of all CTs and the next image shows the chosen CT flows. For the design of a transportation aisle, it can be helpful to display flows based on a transportation unit – as the third image displays containers material flows. These flows are generated based on the transportation network of aisles (see details in Section 4.7.3). The last image shows flows based on realistic shapes of aisles, conveyors and transportation roads.

Table 28 – Magna: results of experiment

Number of facilities – optimal configuration:

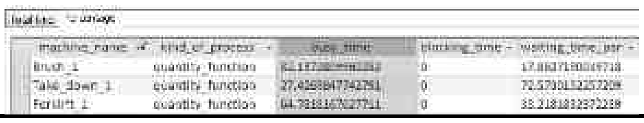
Name	Quantity
Check WS	10
Forklifts	3
Take down	2
JIT WS	14

Throughput:



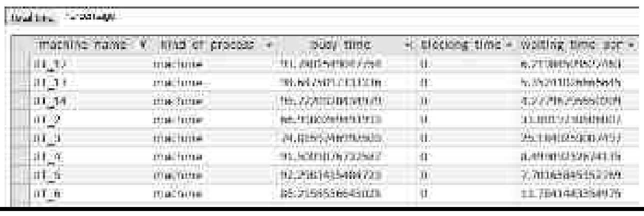
part_throughput	name_throughput	value_throughput
N_Oct_BF_BlackMagic	part_order	[illegible]
Superb_BR_Anthracit	part_order	322
N_Oct_BR_Flamenco	part_order	[illegible]
N_Oct_HSR_Racedit	part_order	[illegible]
Fabia_BF_Anthracit	part_order	[illegible]
Fabia_BR_Sommerblau	part_order	[illegible]
N_Oct_HSR_Cornida	part_order	[illegible]
Superb_BR_Blackmagic	part_order	[illegible]
N_Oct_BF_DiamSilver	part_order	[illegible]
N_Oct_BR_Satin	part_order	423

Facility usage:



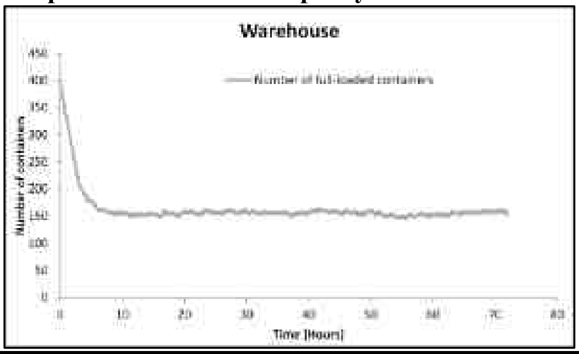
machine_name	kind_of_process	busy_time	blocking_time	waiting_time_per
Brush_1	quantity_function	[illegible]	0	17.8637120019718
Take_down_1	quantity_function	27.4268847742751	0	72.5730132257209
Forklift_1	quantity_function	64.7818167627751	0	33.2181832372239

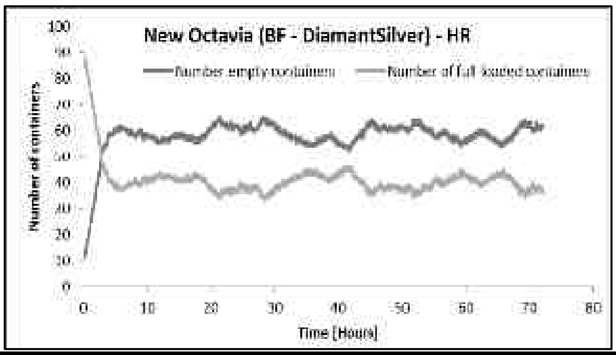
JIT workstations:



machine_name	kind_of_process	busy_time	blocking_time	waiting_time_per
JIT_10	machine	[illegible]	0	[illegible]
JIT_11	machine	[illegible]	0	[illegible]
JIT_14	machine	[illegible]	0	[illegible]
JIT_2	machine	[illegible]	0	[illegible]
JIT_3	machine	[illegible]	0	[illegible]
JIT_4	machine	[illegible]	0	[illegible]
JIT_5	machine	[illegible]	0	[illegible]
JIT_6	machine	[illegible]	0	[illegible]

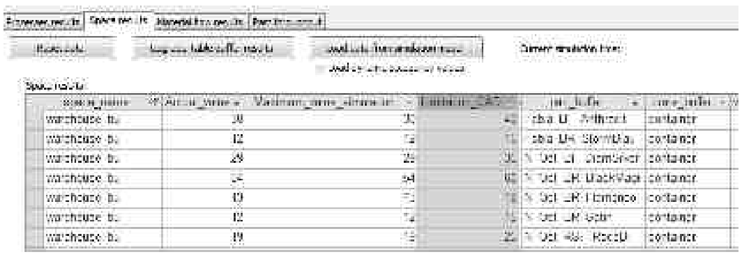
Graphs of containers occupancy in the Warehouse:





b) Buffer size establishment

In order to find the optimal buffer (storages, warehouse) size, it is possible to use the approach described in Section 4.7.3. With this approach, there are integrated results from the simulation *("maximum_value_simulation")* and CAD layout *("limitation_CAD")* as shown in Figure 150. Simulation results provide the maximum number of units (e.g. containers) in the given buffer during the simulated period, and in the CAD layout, there is a specific available space for these amounts of containers. In Figure 150, there are data for the Warehouse with available space provided by the layout for each CT. In this case, there is enough space for all CT containers, as also shown in Figure 151.



[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
warehouse_b.	30	[illegible]	[illegible]	[illegible]	container
warehouse_b.	12	[illegible]	[illegible]	[illegible]	container
warehouse_b.	29	[illegible]	[illegible]	[illegible]	container
warehouse_b.	24	[illegible]	[illegible]	[illegible]	container
warehouse_b.	13	[illegible]	[illegible]	[illegible]	container
warehouse_b.	12	[illegible]	[illegible]	[illegible]	container
warehouse_b.	19	[illegible]	[illegible]	[illegible]	container

Figure 150 – Magna: data

The appropriate number of blocks (oranges coloured blocks) representing containers is generated to the layout (see Figure 151, upper image). Blocks representing 3D containers are automatically inserted into the layout. These blocks are arranged and placed in the correct position into shelves or stacked (see Figure 151, lower image). This is helpful to the design of a warehouse structure, stocking containers in four layers (3D layout), the required manipulation for space (based on material flows and aisles) and the validation of the required space (the maximal number of parts that can appear in the given buffer).

5.3.7 Conclusion of project

This project solves internal logistics in the factory of the automotive supplier, more precisely the logistics between the paint shop (batch production) and the JIT assembly controlled by customers' orders. In order to test the planned increased production (doubled), a simulation model was developed in IDS. With this model, the optimal configuration was established as well as estimated the safety level of containers in the Warehouse. The required time for the complete work in IDS was one and half week compared to 4 weeks needed for solving by traditional approach (see Table 34).

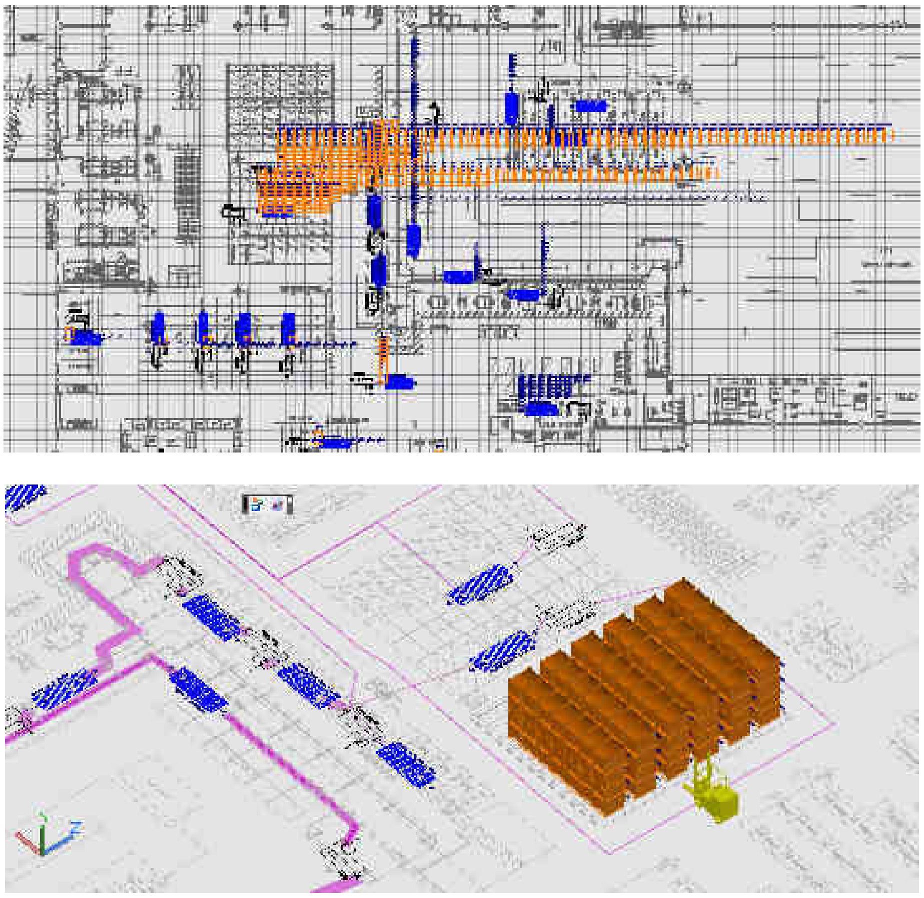


Figure 151 – Magna: 3D layout

5.4 Modus

5.4.1 Project and company overview

Company Modus is a Czech producer of lights for offices, factories, homes, etc. with a monthly production around 120 000 lights of many types and modifications (see Figure 152) (www.modus.cz, 2011).

The company belongs to the greatest light technique producer and exporter of the Czech Republic. The company resulted of the union of several small firms in 1994. It has grown and spread its market activity all over the world.

Figure 152 - Modus: Products

5.4.2 Project overview and re-layout requirements

This project is focused on optimising the production system layout. The reasons are clear – the current layout organisation causes a large material **manipulation**, **transportation** and **storing** of parts that leads to **extra costs** as well as **prolonging lead time**. These factors are typical when re-layout is required.

This department is currently based on the **process layout pattern**, built in the beginning of the company. There was a more specific customer production and the job-shop layout was efficient. Nowadays, the production (product portfolio and volumes) have changed to more stable and the current layout pattern became ineffective due also to the placement of a new kind of production equipment just into free spaces in the factory, not into the optimal position.

Main tasks of this project are concerned in the re-layout activity and change current **process** layout into **product** or **cellular** layouts. In these layouts, production facilities are ordered based on operation sequences or similarities in the production.

Layout alternatives are **designed with** the help of **IDS tools** – PQ analysis, cluster analysis, automatic layout pattern generation and evaluation alternatives based on the total sums of material flows to compare their effectiveness.

This project is solved as simple deterministic model (see Section A.4.8) without involving simulation studies and specific operation. In this case, it is not necessary to analyse and solve it in details; the focus is on the optimization of overall material flows.

5.4.3 Description of technology

This example is focused on metal-forming department facilities with similar technologies. In this department, there are machines and production equipment that are used for:

- cutting raw material (saws, material feeders)
- eccentric presses (getting shapes)
- punching machines
- bending and roll-bending

The production equipment can be moved and then be involved in the re-layout. The following operations and equipment are not involved in this project: painting, mouldering plastic parts, warehouse and final assembling in the workstations (WS). These facilities have fixed position in this project and it is so for the following reasons: the assembly WSs are much closer to the warehouse activities and these two departments must be close. The assembly WSs is convenient in places far from metal-form departments due to the noise, vibrations, etc. Painting facilities consist of a complete line, where moving is very expensive. Mouldering machines need special equipment: a crane to change heavy moulds and also a supply of plastic material.

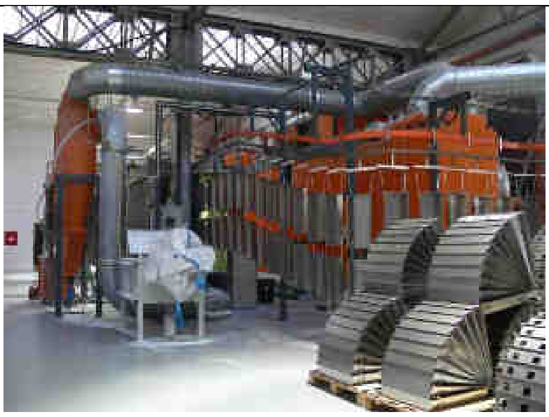
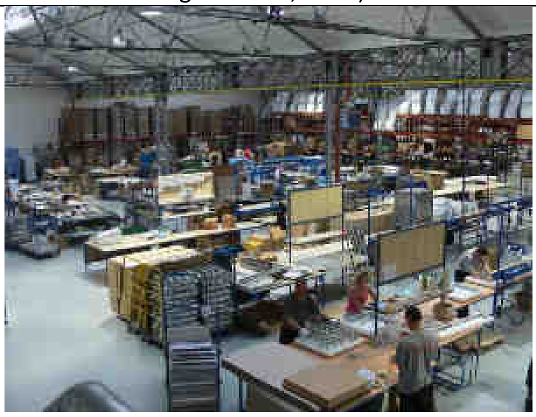
CNC bending machine	**Bending line**
CNC punching machine	**CNC roll bending machine – LUNA** (www.lunabendingrolls.com, 2011)
 **Paint shop**	 **Assembly workstations**

Figure 153 - Images from metal-forming and production areas (www.modus.cz, 2011)

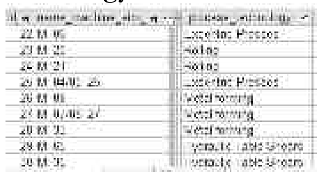
5.4.4 Input data processing

All input data are received from the company's database. Input data are: a list of inventory and used technology, a list of products, production volumes, the current layout and the available space and a sequence of operations. This project involves thirty-four machines and thirty-three products. The light types "I", "LLX", "LLY", "SB", "LL" and "K" are assembled from sub-parts, based on the Bill of material (BOM). The production volume is an average of the week throughput that is based on two years of the production schedule analysis.

This data is saved in DB and AutoCAD.

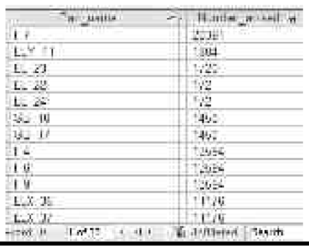
Table 29 – Modus: Input data analysis

List of inventory and technology:



[illegible]	[illegible]
22 M 06	Excentric Presses
23 M 20	Rolling
24 M 21	Rolling
25 M 04/05 25	Excentric Presses
26 M 08	Metal forming
27 M 07/08 27	Metal forming
28 M 33	Metal forming
29 M 12	Hydraulic Table Shears
30 M 35	Hydraulic Table Shears

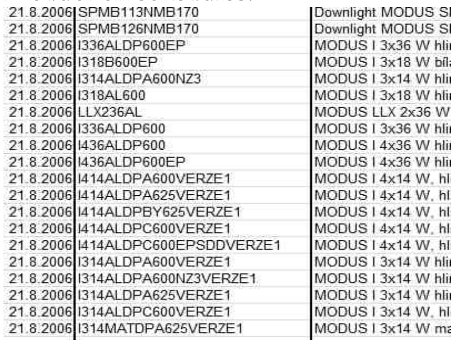
Production:



Part_name	Number_produced
I 7	22081
LLY 11	1804
LL 23	1722
LL 22	172
LL 24	172
SSL 10	1462
SSL 17	1462
I 4	12054
I 8	12054
I 9	12054
LLX 36	1176
LLX 37	1176

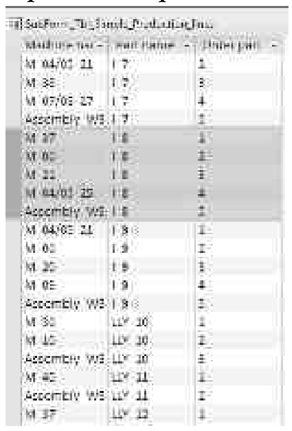
Record: 1 of 13 No Filter Search

Production schedules:



21.8.2006	SPMB113NMB170	Downlight MODUS SI
21.8.2006	SPMB126NMB170	Downlight MODUS SI
21.8.2006	I336ALDP600EP	MODUS I 3x36 W hli
21.8.2006	I318B600EP	MODUS I 3x18 W bil
21.8.2006	I314ALDPA600NZ3	MODUS I 3x14 W hli
21.8.2006	I318AL600	MODUS I 3x18 W hli
21.8.2006	LLX236AL	MODUS LLX 2x36 W
21.8.2006	I336ALDP600	MODUS I 3x36 W hli
21.8.2006	I436ALDP600	MODUS I 4x36 W hli
21.8.2006	I436ALDP600EP	MODUS I 4x36 W hli
21.8.2006	I414ALDPA600VERZE1	MODUS I 4x14 W, hl
21.8.2006	I414ALDPA625VERZE1	MODUS I 4x14 W, hl
21.8.2006	I414ALDPBY625VERZE1	MODUS I 4x14 W, hl
21.8.2006	I414ALDPC600VERZE1	MODUS I 4x14 W, hl
21.8.2006	I414ALDPC600EPSDDVERZE1	MODUS I 4x14 W, hl
21.8.2006	I314ALDPA600VERZE1	MODUS I 3x14 W hli
21.8.2006	I314ALDPA600NZ3VERZE1	MODUS I 3x14 W hli
21.8.2006	I314ALDPA625VERZE1	MODUS I 3x14 W hli
21.8.2006	I314ALDPC600VERZE1	MODUS I 3x14 W, hl
21.8.2006	I314MATDPA625VERZE1	MODUS I 3x14 W ma

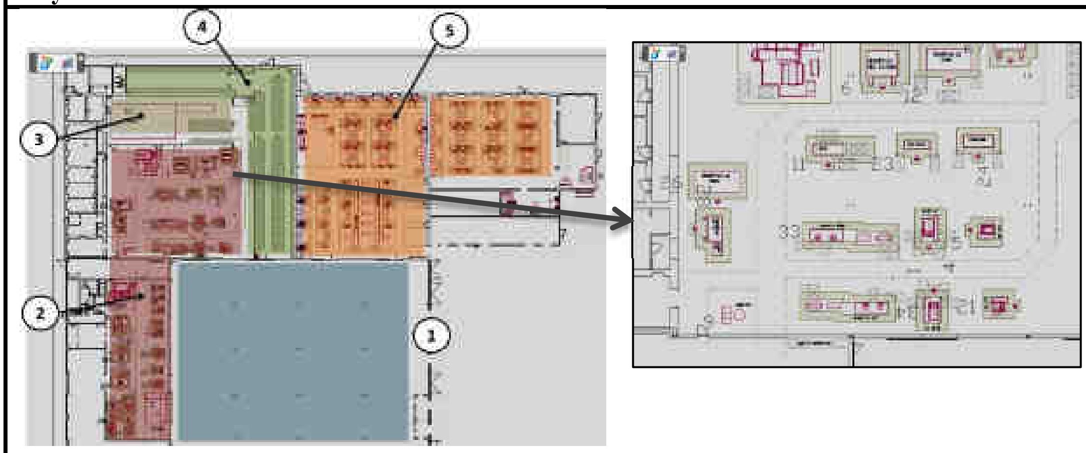
Operation sequences:



SubForm_Tbl_Simple_ProductionPlan

Machine nat	Part name	Order part
M 04/03 21	I 7	1
M 38	I 7	3
M 07/08 27	I 7	4
Assembly WS	I 7	2
M 37	I 8	1
M 02	I 8	2
M 21	I 8	3
M 04/03 25	I 8	4
Assembly WS	I 8	2
M 04/03 21	I 9	1
M 02	I 9	2
M 20	I 9	3
M 05	I 9	4
Assembly WS	I 9	2
M 33	LLY 10	1
M 15	LLY 10	2
Assembly WS	LLY 10	3
M 40	LLY 11	1
Assembly WS	LLY 11	2
M 37	LLY 13	1

Layout

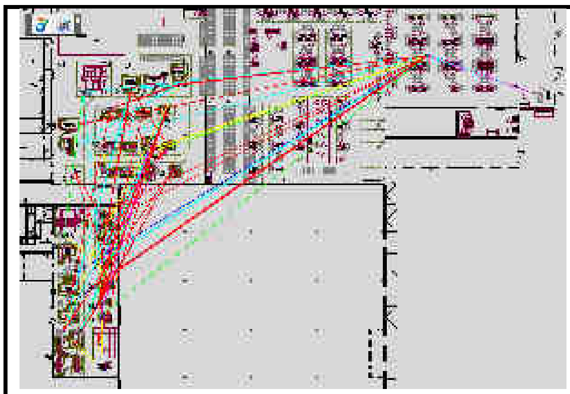
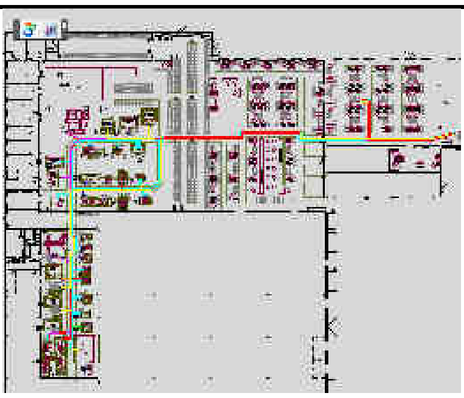
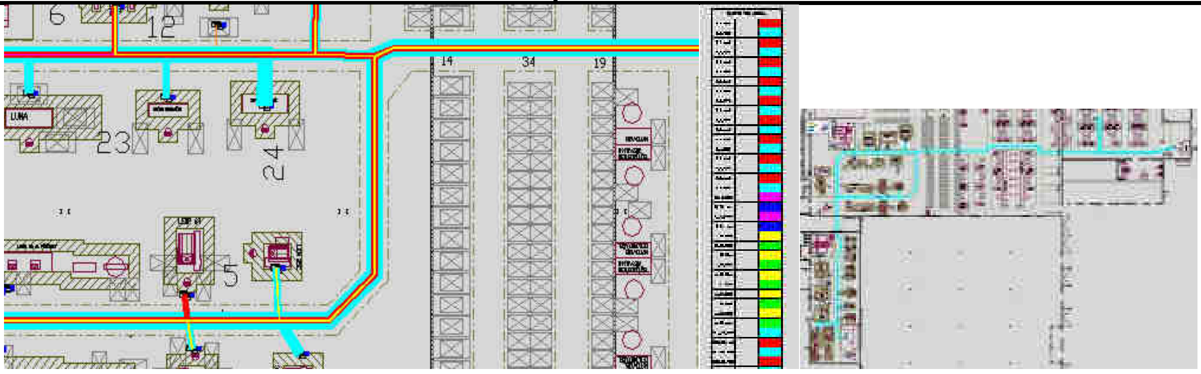
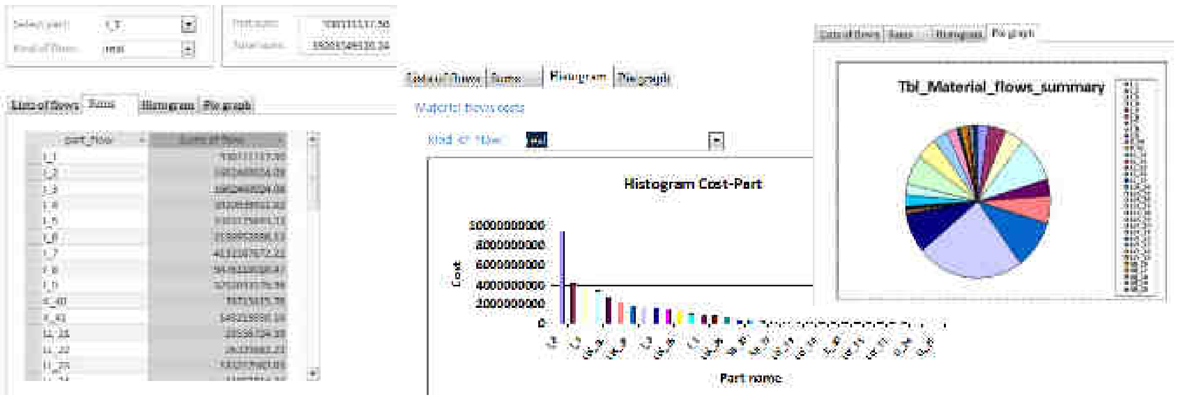


1. **Storage of raw material**
2. **Metal-forming department**
3. **Paint shop**
4. **Warehouse**
5. **Assembly workstations**

5.4.5 Current situation

Based on the input data set, a layout of the current state is made. The resources (list of production equipment) are generated as simple blocks into the layout where the user places them into the correct position on the real machine drawing block. The material flows are generated automatically and the constrained flows are based on the transportation network that is composed of aisles Table 30.

Table 30 – Modus: Current factory layout

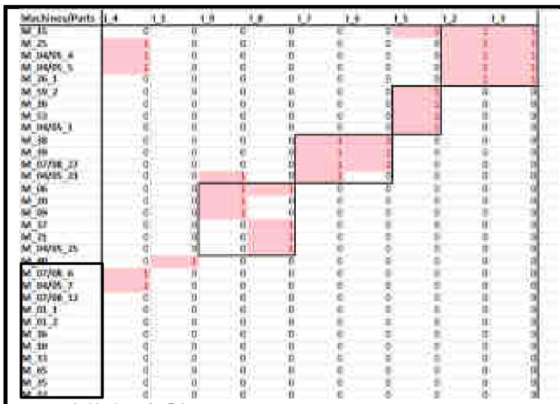
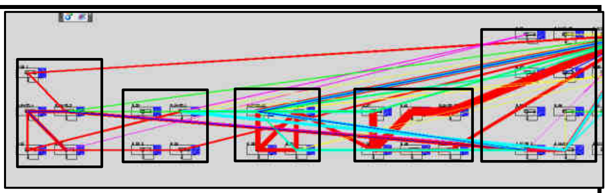
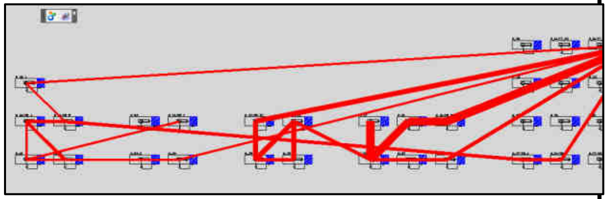
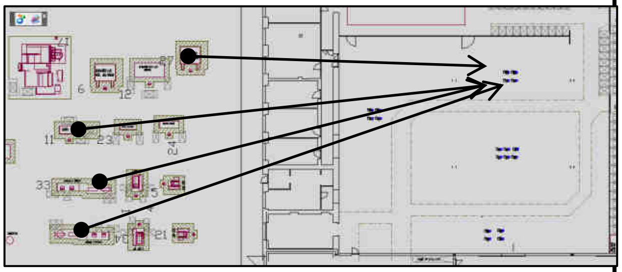
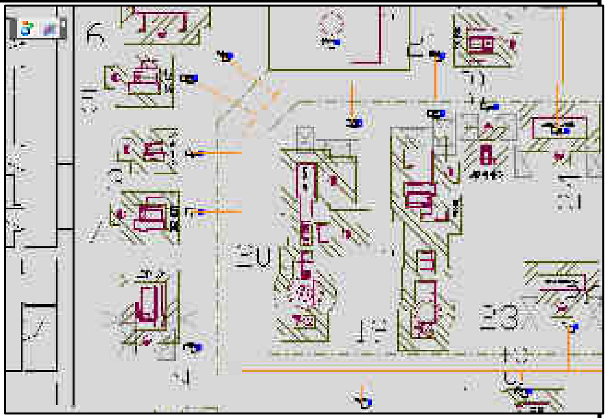
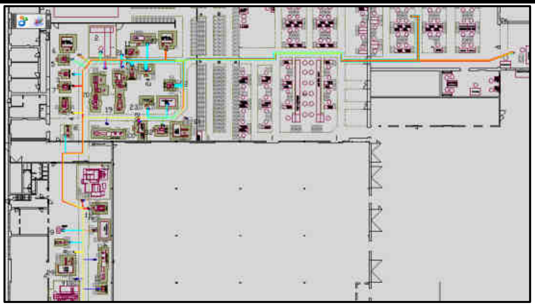
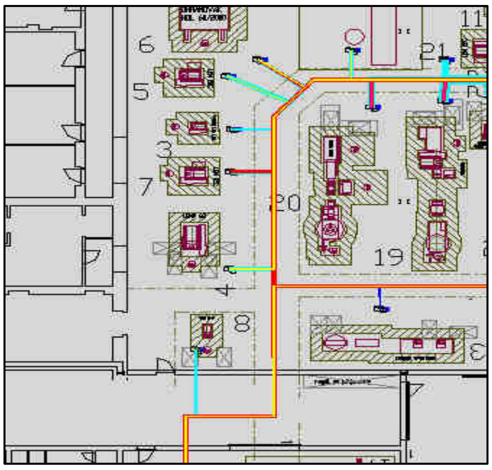
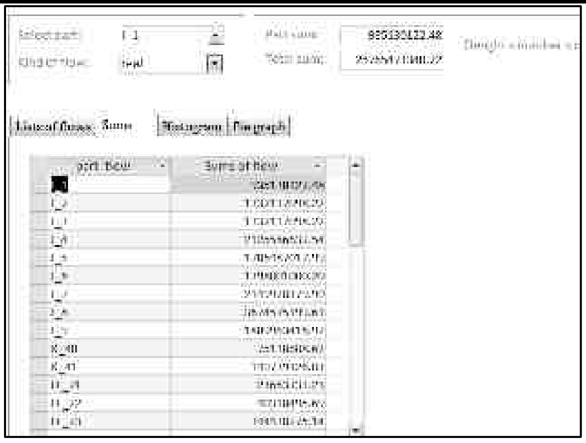
 Layout with ideal material flows	 Layout with constrained material flows
 Layout with transportation flows, legend of material flows and "I" lights flows	
Results:  Data about material flows (CAD polylines) are recorded into DB where sums are calculated. Results can be shown through histograms or pie graphs in order to see the most significant flows. Total costs are 39 203 549 320 (flow lengths in *mm* x products volumes per week x costs per unit).	

5.4.6 Optimised layout – alternative designs by IDS

Based on the results of material flows values of the current situation (**PQ analysis**), it is clear that the most produced parts are the lights "I" and "LLX". Lights "SB", "LLY", "L"L and "K" have very low production volumes (see Figure 154). Machines for "I" parts must be as close as possible to achieve minimum material flows. For the optimization of material flows, the following approaches can be used:

- layout based on clusters (Alternative "Cluster")
- layout based on production lines (Alternative "Production line")

Table 31 – Cluster analysis and layout design process

 **Established Clusters**	  **Automatically generated clusters (black rectangular) and material flows**
 **Blocks of real production equipment arrangement**	 **Design of transportation network based on aisles (orange polylines)**
  **Alternative overview and detail view**	 **Results of "Cluster" alternative**

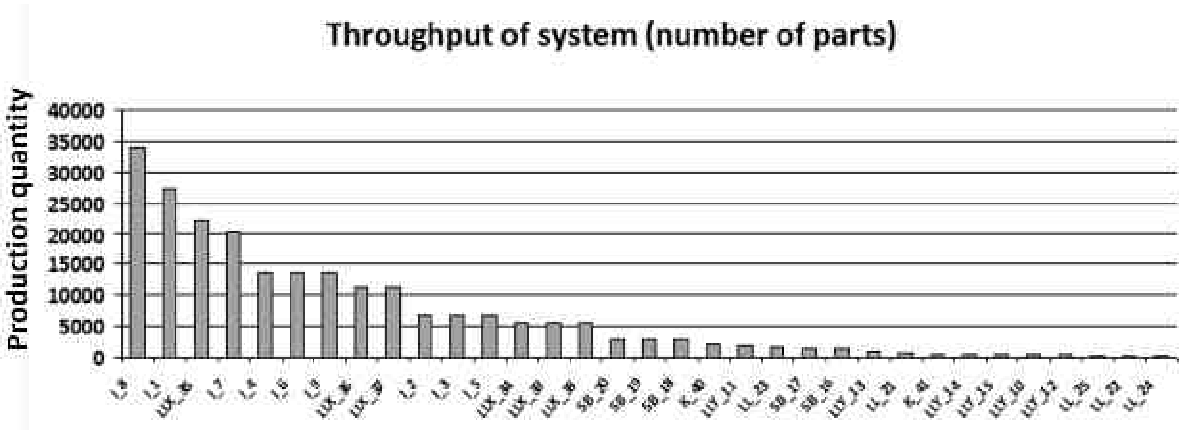


Figure 154 - PQ analysis

a) <u>Alternative “Cluster”</u>

For parts “I” and “LLX” a cluster analysis is made – see Table 31. Based on the cluster analysis, there are four clusters that are identified, and the rest of the facilities are in the exception field. After that, blocks can be generated in the layout as well as material flows. The next step is to arrange the blocks into symbolic blocks of clusters to represent the real production equipment. Between equipments must be space for transportation (transportation network). Based on that, realistic material flows are generated, resulting a total score of 25 765 473 340 which is 65% of the original. This score has unit flow lengths in mm x products volumes per week x costs per unit.

b) <u>Alternative “Production line”</u>

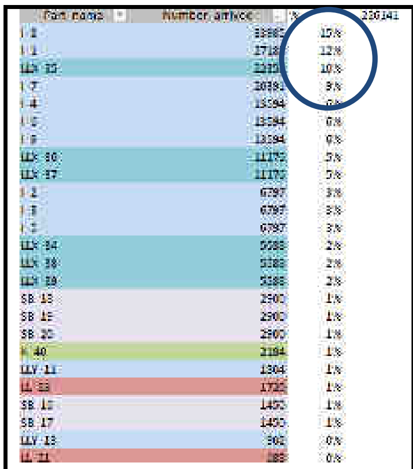
In this alternative, the equipment is placed in the production line by the order of the operation sequence of given parts. It is convenient for the parts with the highest production volumes, as shown in the first image of Table 32. For the three parts, simple blocks of equipment are generated into layouts in the order of the operation sequences. The principle of design is similar to the cluster alternative described in the previous section. Blocks are checked for duplicates that are deleted (e.g. Assembly WS). Then, the rest of the facility blocks are placed and, afterwards, blocks representing real equipment are arranged and material flows generated. The total sum of material flows is 27 320 874 005 that is 69% of the original.

5.4.7 Results and conclusion

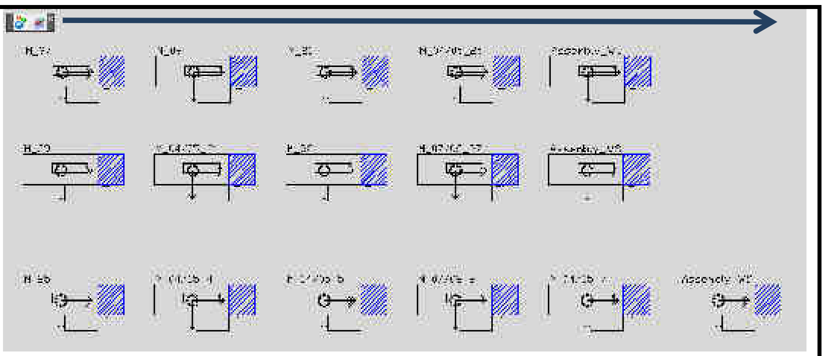
As this simple model has shown, **re-layout** is very important for the current state of production. Two placement methods were introduced: clusters structure and production lines. Both alternatives can be helpful in decreasing the transportation and manipulation activities by around 30%. In both alternatives the percentage is based on the comparison of the total sum of material flows.

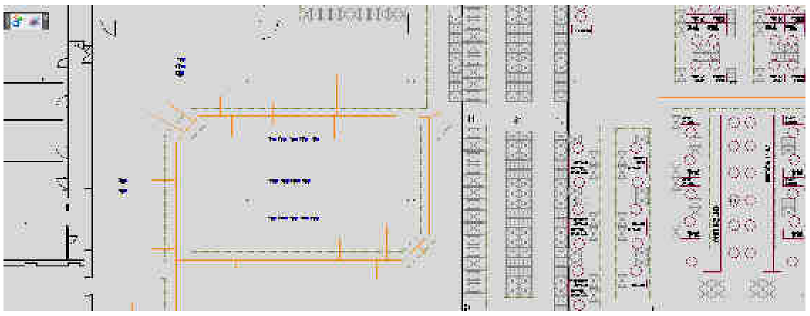
The simulation model and additional experiments done in IDS confirm the use of manipulation equipment and the possibilities for savings.

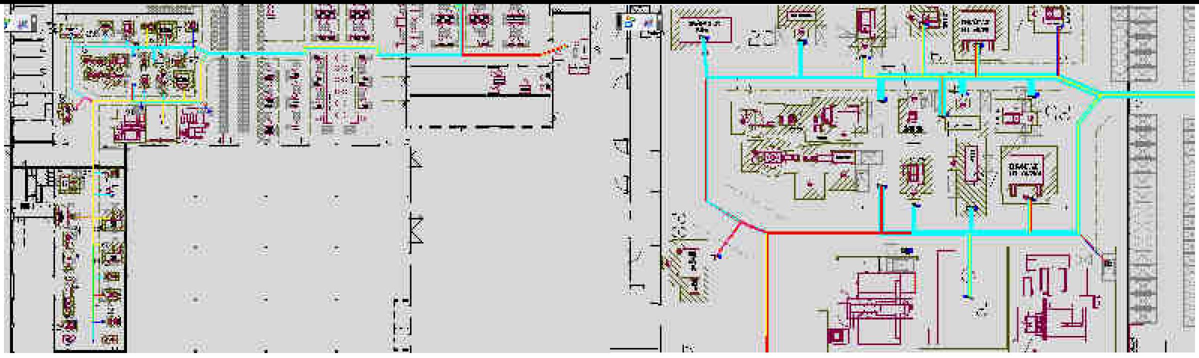
Table 32 – Production line design process



Part name	Number articles	%	236141
I 2	33882	15%	
I 1	27182	12%	
LLX 32	22310	10%	
I 7	20391	9%	
I 4	13394	6%	
I 5	13394	6%	
I 6	13394	6%	
LLX 30	11172	5%	
LLX 37	11172	5%	
I 2	6797	3%	
I 3	6797	3%	
I 2	6797	3%	
LLX 34	5288	2%	
LLX 38	5288	2%	
LLX 39	5288	2%	
SB 13	2900	1%	
SB 15	2900	1%	
SB 20	2900	1%	
K 40	2184	1%	
LLY 11	1304	1%	
LL 33	1720	1%	
SB 15	1455	1%	
SB 17	1450	1%	
LLY 13	302	0%	
LL 31	289	0%	







Alternative B overview and detail view

The suggested alternatives must be discussed with all departments in order to get important feedback, mainly regarding the **re-layout costs,** such as facility movements, design of electricity network, new logistic elements (buffers) and air supply for pneumatic facilities as well as building limitations – possible floor stress, rebuilding inner walls, ergonomics of workstations, etc.

Usually, the biggest problem is to persuade the management to make changes in the current system. It is also very important for the company to make internal scenarios and principles for faster re-layout.

This project was solved by traditional methods, solving with IDS gives similar results (Havlík, Vavruška, & Koblasa, 2006)(Vik, 2007). The difference is that IDS requires around two days of work, while the other methods takes four days. Complete estimated values are in Table 34.

5.5 University campus design

This section shows the principles of factory and facility design on the theoretical example of a university campus design.

Generally, the design of factory layouts is based on relationships between all departments and material or information flows between them (CRAFT, CORELAP and Graph theory algorithms). Factory layouts (campus layout) are composed of block units that, in appropriate number, represent the space size of each department (e.g. Library). This simplification is due to used placement algorithms. The facility layout design is based on detailed information as arrival schedules, operation time or logic control. In this layout, drawing blocks are used with specific shapes of facilities.

This example shows the principle that can be applied to the design of a real factory.

5.5.1 University campus design

This example shows the IDS use in the design of a university campus, similar to Gualtar Campus of UMinho. The main effort of this example is to show the approach of solving.

The aim of this *theoretical* example is to design a layout that covers the established requirements. In this case, it can be minimizing student flows and avoiding the adjacency of some departments, as the library and the sport area.

a) <u>Input data processing</u>

Input data consists of a **relationship chart**, **table from-to** containing flows of students between departments and the **number of students** of each department (see Figure 155). This data is estimated by the author. The "A" in the relationship chart means "absolutely necessary", e.g. the *Institution of literature* adjacent to the *Library*, which allows an easy access to books. On the other hand, "X" relationships mean avoiding placing departments near each other, e.g. *Sport area* and study departments, because of the disturbing noise. The table below contains an estimation of students' movements between departments per day, based on the number of students of each department. The total number of student is 9 300.

As mentioned, layouts are composed of block units that, in appropriate number, represent the size of each department. In Figure 156, there is a table with actual department sizes (areas). They can be obtained with the application Google Earth[1] and AutoCAD for measuring the areas, e.g. the university *restaurant* has an area of around 6400 m^2. One unit (square block) has the following dimensions 25 x 25 m = 625 m^2 and so the number of units is 10 (6400/625 = 10). The dimension of this unit can be different based on requirment of detail level. These input data is recorded to DB, as shown in Figure 157.

[1] Map view with GPS coordination: http://g.co/maps/vtmju

Department name	Number of students
School of Sciences	400
Pedagogical Complex	5000
School of Economics	2000
Institution of Social Sciences	600
Institution of Literature and Humanities	400
School of Engineering	400
School of Medicine	500

Department name	School of Sciences	Pedagogical Complex	School of Economics	Institution of Social Sciences	Institution of Literature and Humanities	School of Engineering	School of Medicine	Library	Restaurant	Technical Services	Sport area	Academic Services
School of Sciences	-	U	U	U	U	E	U	I	O	U	X	O
Pedagogical Complex		-	U	U	U	U	U	E	O	U	X	O
School of Economics			-	U	U	U	U	I	O	U	X	O
Institution of Social Sciences				-	I	U	U	E	O	U	X	O
Institution of Literature and Humanities					-	U	U	A	O	U	X	O
School of Engineering						-	U	I	O	U	X	O
School of Medicine							-	I	O	U	X	O
Library								-	E	I	X	E
Restaurant									-	E	X	O
Technical Services										-	I	I
Sport area											-	O
Academic Services												-

Department name	School of Sciences	Pedagogical Complex	School of Economics	Institution of Social Sciences	Institution of Literature and Humanities	School of Engineering	School of Medicine	Library	Restaurant	Technical Services	Sport area	Academic Services
School of Sciences	0	100	0	0	0	100	0	100	300	12	100	20
Pedagogical Complex		0	500	100	300	100	100	1250	3750	150	1250	250
School of Economics			0	100	0	50	0	500	1500	60	500	100
Institution of Social Sciences				0	100	0	0	500	1500	60	500	100
Institution of Literature and Humanities					0	0	0	500	450	18	150	30
School of Engineering						0	0	100	300	12	100	20
School of Medicine							0	125	375	15	125	25
Library								0	1000	10	500	500
Restaurant									0	20	100	500
Technical Services										0	10	10
Sport area											0	200
Academic Services												0

Figure 155 - University campus: Number of students, relationships chart and from-to table

b) <u>Solution – designed alternatives</u>

As mentioned before, IDS supports the *CORELAP* algorithm for design layout based on the relationship charts. *CRAFT* algorithm and *Graph Theory Block Layout* are based on material flows values. The principle of using and the configuration is described in Section A.4.6.

Name_department	Size department
School of Sciences	8
Pedagogical Complex	20
School of Economics	6
Institution of Social Sciences	5
Institution of Literature and Humanities	5
School of Engineering	6
School of Medicine	6
Library	10
Restaurant	10
Technical Services	3
Sport area	10
Academic Services	2

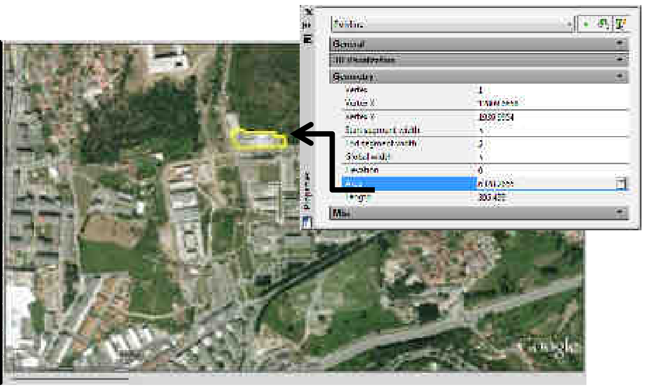
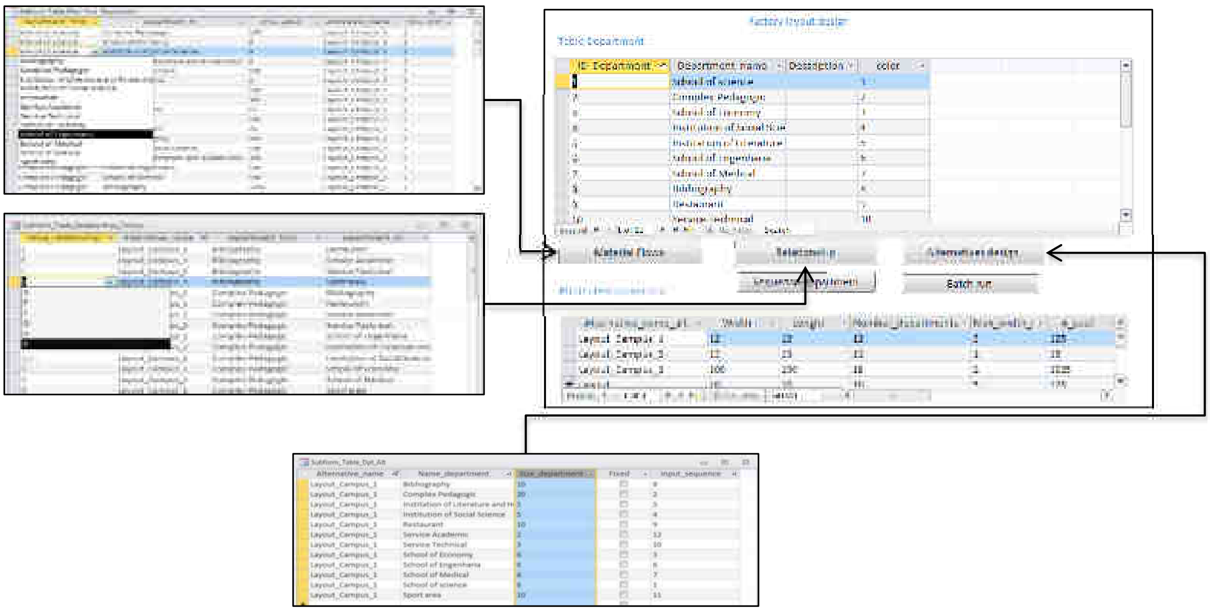


Figure 156 - University campus: Sizes of the departments, snapshot from AutoCAD – measurement of areas



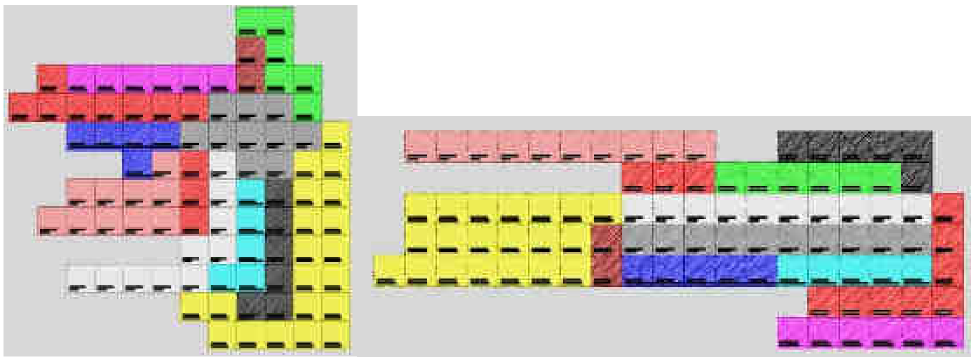
Subform_Table_Dpt_Alt

Alternative_name	Name_department	Size_department	Fixed	Input_sequence
Layout_Campus_1	Bibliography	10	☐	8
Layout_Campus_1	Complex Pedagogic	20	☐	2
Layout_Campus_1	Institution of Literature and H	5	☐	5
Layout_Campus_1	Institution of Social Science	5	☐	4
Layout_Campus_1	Restaurant	10	☐	9
Layout_Campus_1	Service Academic	2	☐	12
Layout_Campus_1	Service Technical	3	☐	10
Layout_Campus_1	School of Economy	6	☐	3
Layout_Campus_1	School of Engenharia	6	☐	6
Layout_Campus_1	School of Medical	6	☐	7
Layout_Campus_1	School of science	8	☐	1
Layout_Campus_1	Sport area	10	☐	11

Figure 157 - University campus: Database tables and control user forms

CORELAP based layouts

A lot of different alternatives can be generated based on parameters setup (values of AEIOUX, full edge neighbouring/corner neighbouring, kind of algorithm or limitation of initial factory space). In Figure 158, there are examples of generated alternatives and their legend.



	A	B	C
1	Department Legenda		
2	School of science	8	🟥
3	Complex Pedagogic	20	🟨
4	School of Economy	6	🟩
5	Institution of Social Science	5	🟦
6	Institation of Literature and Humanistics	5	🟦
7	School of Engenharia	6	🟪
8	School of Medical	6	⬛
9	Library	10	◼
10	Restaurant	10	⬜
11	Service Technical	3	🟥
12	Sport area	10	🟥
13	Service Academic	2	🟫

Figure 158 - University campus: alternative layout based on CORELAP

CRAFT layouts

Examples of automatically generated alternatives are in Figure 159. IDS provides the display of generated layouts in each improvement step. In each step, two departments that are exchanged to decrease the total flows between departments' centroids. The optimal layout is the last layout from Figure 159.

The aim of this example is to minimize the movements. For design layouts, a map from Google Earth is used as background to place department blocks easily. Figure 160 shows the **current campus** situation (left image) and the optimal theoretical solution conceived by CRAFT with the departments arranged in coloured blocks. It does not consider urbanism, architecture requirements or other constrains; all buildings are placed as close as possible. In order to compare, there is a score value used that is the total sum of flow's length multiplied by the appropriate number of movements per day (unit = meter * # student movements / day). The current campus has a layout score of 23 348 690 (while the theoretical layout is 9 576 830).

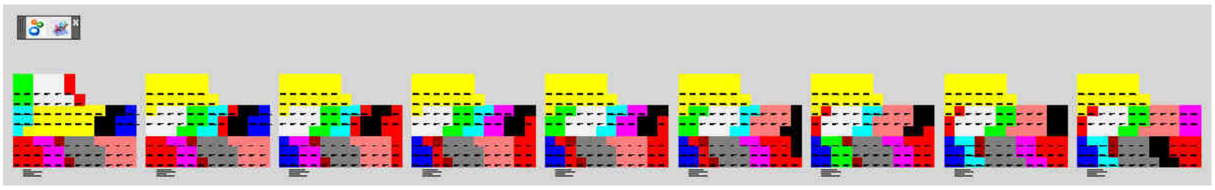


Figure 159 - University campus: alternatives layout based on CRAFT

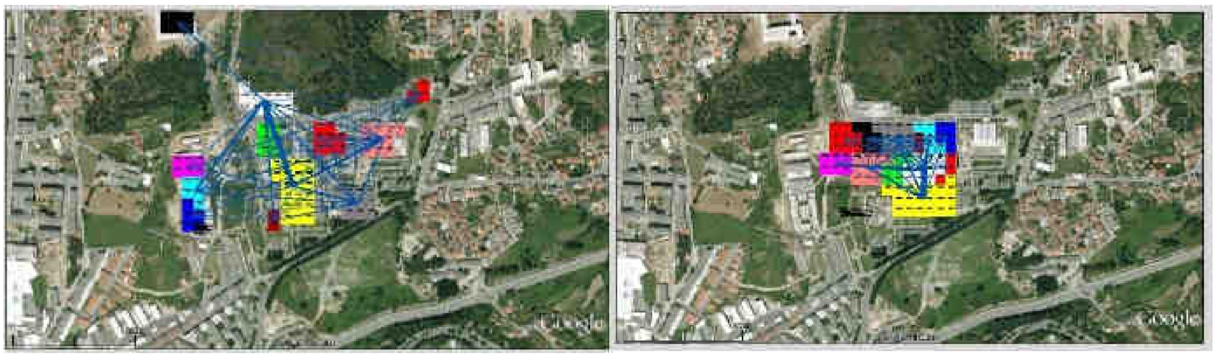


Figure 160 - University campus

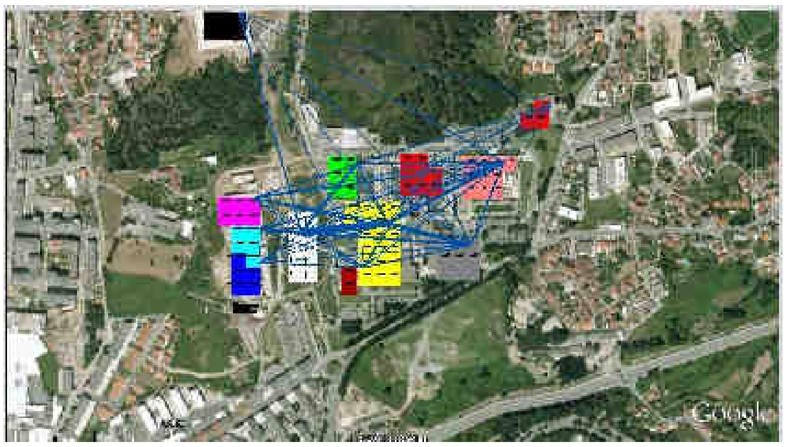


Figure 161 - Improved university campus

Improving the current layout – change of the restaurant position

The current layout can be theoretically improved by moving the university *restaurant* (white blocks) closer to the *Pedagogical department* (yellow blocks) due to the large material flows (see current layout on the previous figure). The optimal position place can be in unallocated space between the departments' buildings. The layout score of the improved layout is 19 772 840, that is 85% of the current layout score. This value can be taken for savings.

For emphasize the benefit of this possible saving, it can be calculated as time saved by students. Based on the score of the current layout, it means that every student goes, in average, 2126 metres every day (score divided by the total number of students). It is 35 minutes spent walking (walking speed = 1 meter per second). After applying the proposed solution, saving can be of 373 metres (6.2 minutes) per day. This saved time can be used for a more productive activity (e.g. studying); moving and generally transportation are time-wasting activities. One cheap hour of work costs 5€. One student would save 0.52€ per day (103€ per year with 200 days). The total amount of money saved of all students can be 965 000€. This calculation does not consider re-layout costs, i.e. building a new restaurant, as well as moving student in the transportation road net, etc.

Savings are abstract values in this example. In a real industrial case, these kinds of savings are important and fundamental; mainly in the use of transportation system (forklifts, milk-run, buses, etc.) and it must be included in the phase of layout design.

c) <u>Conclusion</u>

In this section, a principle of design layouts was shown by the use of CRAFT, CORELAP and other tool in IDS on the theoretical example of the university campus. The principle of factory design is the same in the practical cases. The phase of factory design is very important for the overall system performance and efficiency. Selecting the optimal alternative is the key issue of this phase. The IDS system generates a lot of alternatives from which users can select the optimal and the most effective, avoiding a future expensive re-layout.

5.5.2 University restaurant

This model follows the previous one (design of the university campus) and it is focused on the design of one chosen department – a restaurant for students ("Cantina"). It is a theoretical example that shows the possibility of work in IDS and the procedure from general to detailed models.

The specific aim of this model is to evaluate, by simulation the waiting time constrained by space (number of tables).

The campus layout from the previous model (Figure 160) is used with a defined size. The space for the restaurant is 10 square units, which are 6250 m^2. Only 4 units (2500 m^2) can be occupied with tables (area for eating), while others are reserved for kitchen, administation, storages of food, washing dishes, etc.

a) Conceptual model

A simple model is displayed in Figure 162. Students arrive at the "Cantina"; they take cutleries, drinks and food. Then, their tickets are checked and they can choose a table. After eating, they leave the building. Each table can have 1 to 4 students.

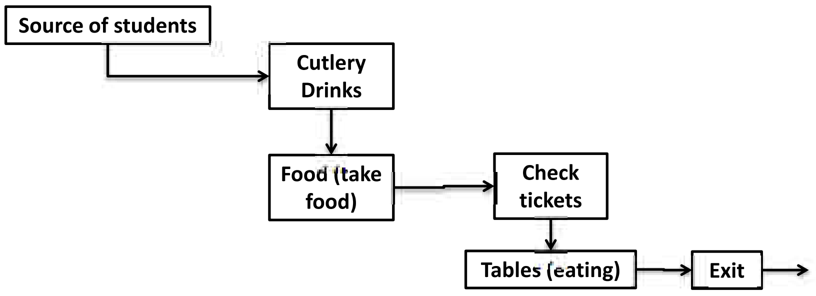
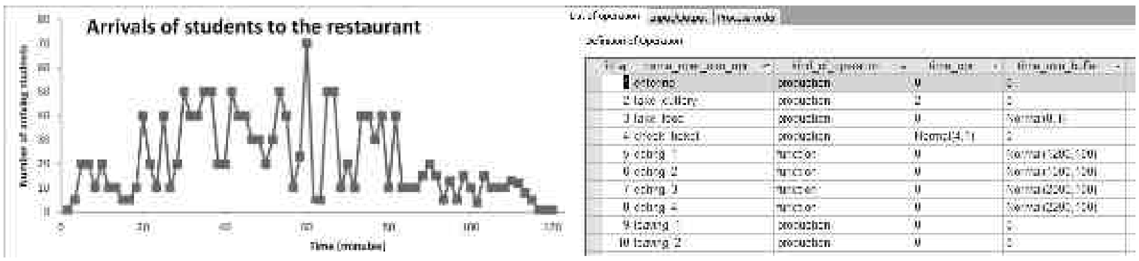


Figure 162 - Cantina: Conceptual model

b) Input and processing data

Input data consists of arrival schedules of students, which is shown in the graph in Figure 163, left image). These values are just estimations. In the right image, there is a list of operation. The time for eating is based on the number of students. Bigger groups need more time to eat, and so this table is occupied for a longer time because students usually wait for each other. The Cantina is open for 2 hours per day; in simulation it is from 0 to 7200 seconds.



id_op	name_type_oper	kind_of_operation	time_op	time_unit_buffer
1	entering	production	0	5
2	take cutlery	production	2	5
3	take food	production	0	Normal(0,1)
4	check ticket	production	Normal(4,1)	2
5	eating 1	function	0	Normal(1200,120)
6	eating 2	function	0	Normal(1200,120)
7	eating 3	function	0	Normal(2200,120)
8	eating 4	function	0	Normal(2200,120)
9	leaving 1	production	0	5
10	leaving 2	production	0	5

Figure 163 - Cantina: Input data

c) Experiment and results

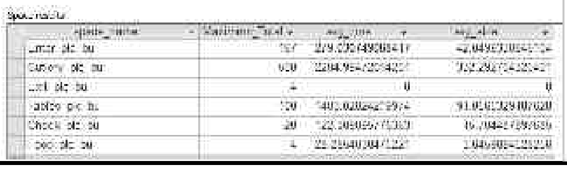
There were 3 experiments processed for the parameter "number of tables" = 100, 200 and 135 (actual Cantina size).

From the graph of *"Number of occupied tables"* and the results tables, it is obvious that 100 tables is too little (which causes a very large waiting time – in average around 2200 seconds – 36 minutes). 200 arranged tables decrease the average waiting time to 8 minutes. However, there is a space limit and this amount of tables is not possible in that place (see Figure 164, right image). In Figure 164 (left image), there is the maximum number of tables that is possible to be arranged, considering aisles between them.

Table 33 – Cantina experiment results

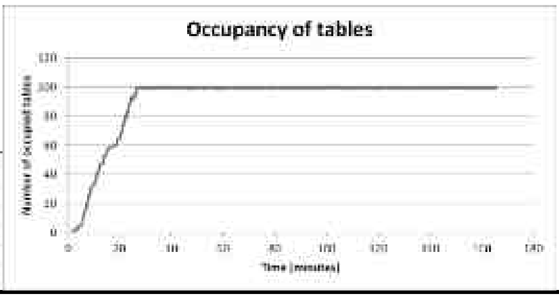
Number of tables
Waiting time in queues Occupancy of tables

Number of tables = 100

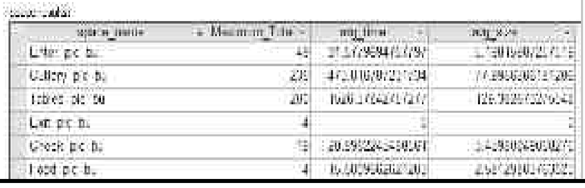


Space results:

space_name	Maximum_Tide_Size	avg_time	avg_size
Enter_pic_bu	157	279.292749286417	42.0492702245704
Cutlery_pic_bu	620	2204.95472204221	132.292704322427
Exit_pic_bu	4	0	0
Tables_pic_bu	120	1493.02824279974	91.0161329107020
Check_pic_bu	20	122.308595775303	16.7044273978639
Food_pic_bu	4	23.2264030847222	1.8469094202210

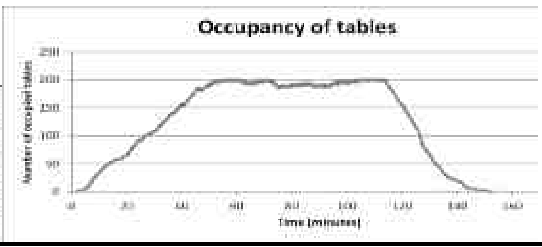


Number of tables = 200

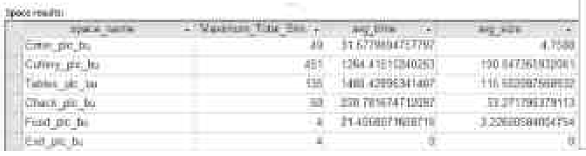


Space results:

space_name	Maximum_Tide_Size	avg_time	avg_size
Enter_pic_bu	45	21.1779694727797	1.7301050221715
Cutlery_pic_bu	273	472.0107072211734	77.8908208131206
Tables_pic_bu	200	1026.17642717217	126.902870327624
Exit_pic_bu	4	0	0
Check_pic_bu	15	20.8992243458087	3.42980249030272
Food_pic_bu	4	15.6095626212013	2.5872901075025

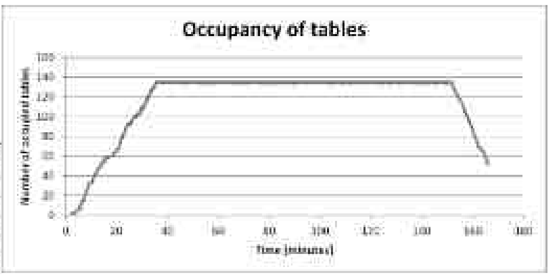


Number of tables = 135



Space results:

space_name	Maximum_Tide_Size	avg_time	avg_size
Enter_pic_bu	49	31.6779894757797	4.7580
Cutlery_pic_bu	451	1264.41810840253	100.847261932061
Tables_pic_bu	135	1488.42898341407	116.550097568032
Check_pic_bu	30	239.781674712897	33.271796379113
Food_pic_bu	4	21.4058071638719	3.22608584054754
Exit_pic_bu	4	0	0



d) CAD layouts

Figure 164 shows layouts of "Cantina" with automatically generated material flows and also automatically generated blocks representing tables.

e) Conclusion

There can be many solutions for this system depending on the Cantina's size. The current Cantina space enables providing 135 tables (see Figure 164, last image) and consequent waiting time of 21 minutes (see results in Table 33, third experiment). If one more unit (625 m^2) of space is provided for tables (200), the evaluated benefit for students is reducing waiting time by 14 minutes (21 - 7).

If there are 1500 students per day, this means 21 000 minutes (350 hours) wasted by waiting in the queues. At 5 € / hour, it means 1750€ per day (350 000 € per year).

This innovative approach, using IDS, enables the calculation of the actual value of one space unit to this subsystem.

This example shows the integration in the different levels – factory (departments design) and facility level (resources and equipment in the department) as well as feedback in both levels.

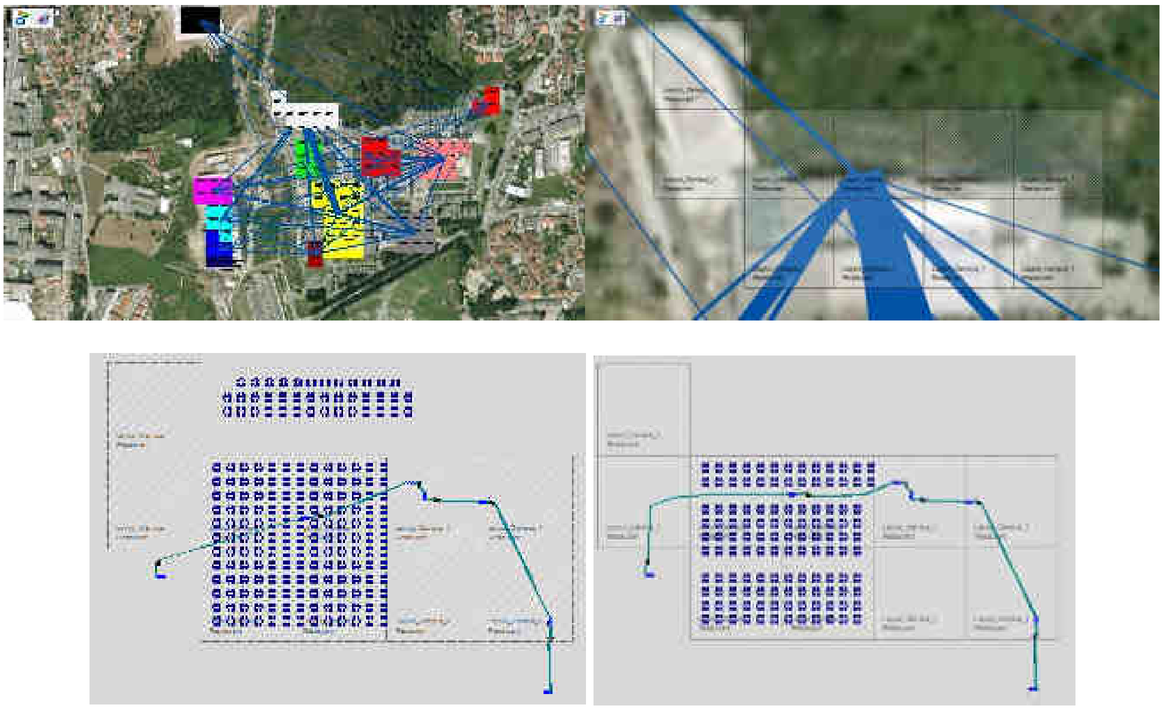


Figure 164 - Cantina: layouts (campus layout, detail of restaurant, layout with 200 tables and layout with arranged tables)

5.6 Summary

Five examples were introduced, from different activity areas. These examples were solved by IDS and also by traditional approach (e.g. manually created simulation model). Table 34 summarises required times for each project, based on estimation. First column for each project means time for traditional approach, second column for IDS. Values in bracket are not involved in the total time due to absence in the one approach.

The conclusion of the practical use of IDS and the IDS approach is summarised in the next chapter – conclusion.

Table 34 – Time requirement summary

Project/ Methods	**Cachapuz**		**Magna**		**Blaupunkt Bosch**		**Modus**	
Traditional approach/IDS								
Project aims establishing	8 hours	8 hours	2 days	2 days	3 weeks	3 weeks	4h	4h
Data analysis and input to DB (IDS)	2 weeks	2 weeks	3 days	3 days	1 week	1 week	8h	8h
Simulation model building	3 weeks	5 minutes	2 weeks	5 minutes	3 weeks	5 minutes	-	-
Experiments /alternatives design	1 hour	4 hours	3 hours	8 hours	3 days	1 days	-	-
CAD layout 2D	1 day	3 hours	2 days	3 hours	2 day	2 hours	2 days	3h
CAD layout 3D	3 days	1 day	-	1 hour	(1 week)	-	-	-
3D animation	(3 weeks)	-	-	-	(4 weeks)	-	-	-
Results explanation, evaluation and documentation	8 hours	1h	8 hours	1h	3 days	8 hours	4h	1h
Total time	6 weeks	3 weeks	4 weeks	1.5 weeks	9 weeks	5 weeks	4 days	2 days

6 Conclusion

Integrated Design of Systems (IDS) was introduced and main principles were discussed. IDS was tested on several small theoretical examples throughout the work, but also tested through 5 real size practical projects. In this Chapter, the mentioned challenges are discussed once they finished up proving to be the main contributions of this thesis. IDS still shows some weaknesses, also presented, together with a few ideas so as to improve IDS in the near future.

6.1 Main Contributions

IDS was tested in many different practical examples:

- general factory design (**University campus**) and its department (restaurant),
- re-layout of a job-shop production based system (**Modus**),
- planning of material flows and implementation of the logistic system (**Cachapuz**),
- internal logistic system in the automotive industry (**Magna**),
- design of an internal supplying system – milk-run (**Blaupunkt Bosch**).

In all these application areas, IDS has proved to be a better solution than traditional approaches - providing faster and better solutions. Also it proved to be an approach that causes less errors – either due to data transfer actions between different software applications or due to bad design decisions. Furthermore, these eventual error situations can easily be detected through the implemented visual functions. Also, IDS makes use of a unique database for any type of project, through a common data structure and format for all projects.

The main contributions are summarised below, with a focus on the importance of the production tools used, on the generic type of application developed, and on the effectiveness of the solution achieved.

Design of a compact integrated system involving CAD and simulation systems

As explained above, the Database system proposed should provide an open structure, allowing integration and data exchange between the simulation tool and the CAD system. These are, in fact, two important tools as far as the production system performance is concerned. Thus, if it is possible to integrate contributions from both tools, the design of a new or an existing production system would gain both in terms of efficiency and effectiveness.

To accomplish this aim, an innovative software tool was designed so as to permit a full integration between CAD and Simulation - changes in the production system configuration managed in one of the applications would take effect in the other application. This means that results from the simulations are used to improve CAD layout designs, and CAD layouts data are inputted in new simulation experiments. This process should work iteratively so as to take advantage of the integrated approach. Furthermore, IDS allows the automatic generation of simulation programs and the automatic generation of CAD layouts. This is, really, a strong contribution to industrial community and a powerful message towards an effective use of these two important production tools – AutoCAD and WITNESS simulation tool were the chosen tools for this important purpose (MS Access was the Database system used).

General approach

As far as IDS is concerned, the general approach adopted has several different meanings – each of these playing a different role towards the innovative contribution of the software tool developed. Thus, the mentioned general approach would enable:

- The possibility of using IDS on different phases and different levels of the design process
- The possibility of using IDS to support design of factories but also to support facility layouts
- The possibility of using IDS as a generic tool (for example to create a base model), but also as a very specific tool (based on the previous model and improving it) for very specific and detailed design problems
- The possibility of using IDS in different types of industrial application areas, but also in the service sector

Effective design processes

IDS also means a set of developed functions that do support a high level of automation in some design phases - the most relevant functions are related to production performance analysis (database), to automatic generation of simulation models (simulation), to transportation, to the generation of basic layout patterns and factory placement methods (CAD), to material flows display in the layout, to transportation network design, finding shortest path between resources, to the 3D drawing blocks library, and to the iterative buffer sizing process.

These functions dramatically decrease required time for the design process when compared with work performed manually (for example, the usual process of creating a simulation model involves a few days, while only a few minutes are needed for the automatic generation of simulation models).

A serious risk as far as IDS concept is concerned, mainly due to automatic type of procedures (automatic generation of simulation models, automatic generation of layouts) would be related to the possibility of using IDS only in very specific type of industrial tasks. However, much concern was dedicated to the development of IDS functions, trying to end up with suitable functions for different application areas and to multiple industrial tasks. In fact, Chapter 5 (on IDS implementation), has proven the adequacy of IDS for the industrial sector (different industry fields) and for the services sector – dealing correctly with a great variety of control logic and operations environments (e.g. assembly, unpacking, waiting rules, etc.).

6.2 Weaknesses

Proposed solution involves several drawbacks and weaknesses. Some of them were supposed due to IDS concept and described in Section 3.2.6, some of them comes from IDS implementation and practical usage. Supposed drawbacks of this solution were known before implementation and there was an effort to avoid of them.

6.2.1 Concept weaknesses

The main issue concerning eventual weaknesses of the solution proposed is that in most cases problems were already identified and, thus, the implementation process has evolved accordingly.

One of the relevant weaknesses is related to input **data preparation** and loading into the Database. As mentioned before and proved on the practical problems experimented, each company uses different data structures and different data sets. This in fact means that there must be a preliminary effort on preparing and formatting data (for example in MS Excel) – this way the data loading phase (to the database) would be easily accomplished. This type of approach would be more efficient when source data is available from an Enterprise Resource Planning (ERP) system. When this is not the case, and source data is not available in digital format, then the developed IDS input forms must be used.

One of the common drawbacks of **background simulation** is the usual poor graphic features associated to generated models - due to the main focus being the automatic generation process, rather than graphic capabilities of the tool used. However, simulation animation could be crucial as to understand the production system behaviour but also for presentation purposes and for the communication with the problem owner. IDS also contributes to the improvement of this issue – all simulation objects can be generated and easily placed in the simulation model – interacting with CAD layout positioning, increasing simulation model logic and behaviour perception (see Section A.4.3.3).

Another weakness is associated with required **time for running** simulation model experiments. Compared to manually made models that are focused only at specific given problems, automatically generated models solve complete set of tasks during simulation running and simulation (calculation) needs more time. As shown in Table 34, time needed for all experiments can be three times higher.

Often, the description of a production system tends to be simplified. For example, IDS does not take into account detailed description of the **human resource behaviour** – there are no functions dedicated to this aspect of the production system. This corresponds to a detail not included in the IDS system, once it would turn too complex the automatic generation of simulation models process.

The generation of facility layout alternatives, being a fully automatic design phase, could lead to some difficulties as far as object identification and interpretation of results is concerned. Thus, all generated objects would be attached with a correct description of the object. Results would then be explicit, including the most relevant information associated – with appropriate graphs and tables. Also great attention has been dedicated to input/output control forms.

Dealing with lots of developed functions would imply that IDS simulation models would also involve a large number of internal variables and internal objects. This means that these specific objects would only be used for simulation purposes – in fact, they do not actually belong to the real system, making use of these variables and objects could be a difficult task for the user.

6.2.2 Difficulties experienced in real applications

As far as the actual application of IDS through a set of real projects, some difficulties were experienced.

In "Cachapuz" project it was found that IDS would improve if it were possible to have a fully automated experimentation step for predefined set of parameters, variables and object properties (see Table 24) – avoiding permanent action from the end user.

In "Blaupunkt" project, setting a milk-run system and respective operation could be confusing, despite the great effort on this issue. Thus, correspondent user form could be improved.

In "Magna" project, it was found that for high complex problems with huge number of operations would significantly increase simulation running time.

6.2.3 Difficulties experienced during IDS development

One of the most problematic parts in the IDS development was the automatic generation of simulation models due to complete absence of some additional material on description of WITNESS objects and properties.

In AutoCAD, there are some problems in the speed of generating some specific objects such as tables (e.g. legend tables for material flows). While using VBS programming, some stability problems concerning the programming interface in AutoCAD emerged. Some problems appeared due to using of VBA programming due to stability of the programming interface in AutoCAD.

Also, AutoCAD does not allow the user to assign a specific name and other IDS properties to each type of AutoCAD objects. Nevertheless this concern, IDS would overcome it using an indirect type of solution - through the use of blocks, references or grouping objects.

Also for projects involving huge data sets, time to load data from WITNESS or AutoCAD is much longer.

6.3 Future work

Future work can be focused on several issues; firstly current weaknesses coming from real applications experience, implementation or even conceptual weaknesses could be overcome and IDS improved. Then, other issues would concern the development of new IDS functionalities.

As mentioned, one of the weaknesses is higher required **time for running of simulation** experiments compared to manually made models. This can be improved if IDS would allow easy functions and variables launching (selected by user). That could help to decrease required time for simulation running. IDS already includes a basic setup for control of some functions during simulation running – nevertheless this only occur internally, with no user interface.

As mentioned above, IDS is an open software tool and, in this respect, it is possible to add new functionalities or even to change existing functionalities, improving the tool and, possibly, making it even more suitable to some specific application areas.

In fact, a full integration of an enterprise database system (an **ERP system**) with IDS would be beneficial. This means that the possible development of a universal data integration interface, enabling an easier process of interpreting and loading production system data to IDS platform would be of great relevance. If this is achieved, then IDS could reach a stage so as to be considered as a Digital Factory System.

One of the interesting issues can be automatic preparing data about production system directly from some draft exercise, e.g. using **MS VISIO** or **MS PowerPoint**. This would also be helpful for filling the gap between concept model and database model.

As already stated, the work under **automatic running of experiments** can be significantly improved. Some special input form would be developed where user setup experiments and parameters would be loaded. Results from simulation experiments would then be loaded into the DB. This functionality could also require full **automatic documentation**. Thus, reports involving tables of results, graphs etc. could be generated, e.g. using MS Access report objects.

Graphics characteristics of IDS could also be improved. For example, animation of objects could be included – physical movements. In this respect, **WITNESS Virtual Reality Module** could be used. This module provides construction of realistic 3D graphics from 2D simulation model - replacing 2D icons by 3D drawing objects. These objects can be used directly from AutoCAD 3D layout or even from internet library of free 3D models - **Google Warehouse**.

WITNESS simulation tool also contains a tool for **optimization** of the given system through a set of optimising methods (Simulated Annealing, genetic algorithm). Thus, optimum transportation policies or optimum production batches could be found as well as sequence for jobs, number of resources (equipment), etc. Objective functions and parameters should then be determined and used.

Layout designing can involve also automatic arrangement of drawing blocks representing objects with real-shape not only simplified blocks.

Another possible improvement for IDS would involve the way to program in AutoCAD. Replacing VBA with .NET programming technology would be of much interest, as IDS would provide completely new objects and this would give more interactivity to the design, animated objects, etc.

References

Complete reference data can be founded also in web page: http://www.mendeley.com/groups/1716055/references/. For references, “Chicago” referencing style rules are used.

Adusumilli, K.M., and R.L. Wright. 2004. Comparative factory analysis of standard FOUP capacities. In *Proceedings of the 2004 Winter Simulation Conference*, 2:1930–1934. IEEE Computer Society Press.

Ahmad, A.R., O.A. Basir, K. Hassanein, and M.H. Imam. 2004. A Placement Algorithm for Efficient Generation of Superior Decision Alternatives in Layout design. In *Proceedings of the 5th International Conference on Operations and Quantitative Management (ICOQM 2004)*. Seoul.

Aleisa, Esra, and Li Lin. 2005. For effective facilities planning: layout optimization then simulation, or vice versa? In *Proceedings of the Winter Simulation Conference*, 1381-1385. IEEE Computer Society Press.

Alexander, Michael. 2007. *Microsoft® Access^{TM} 2007 Data Analysis. Analysis.* Indianapolis: Wiley Publishing, Inc.

Altinkilinc, M. 2004. Simulation-based layout planning of a production plant. In *Proceedings of the 2004 Winter Simulation Conference*, 2:1079–1084. IEEE Computer Society Press.

Anstreicher, K., N. Brixius, J.P. Goux, and J. Linderoth. 2002. “Solving large quadratic assignment problems on computational grids.” *Mathematical Programming* 91 (3): 563–588.

Apple, James. 1972. *Plant Layout and Material Handling.*

Askin, R. G., and M. Zhou. 1998. “Formation of independent flow-line cells based on operation requirements and machine capabilities.” *IIE transactions* 30: 319–329.

Baker, R I, and P G Maropoulos. 1997. “An automatic clustering algorithm suitable for use by a computer-based tool for the design , management and continuous improvement of cellular manufacturing systems.” *Science* I (3): 217-230.

Banerjee, P., B. Montreuil, C. Moodie, and R. Kashyap. 1992. “A modeling of interactive facilities layout designer reasoning using qualitative patterns.” *International Journal of Production Economics* 30 (3): 433–453.

Banks, Jerry, John Carson, Barry Nelson, and David Nicol. 2005. *Discrete-event system simulation.*

Beaulieu, A. 2009. *Learning SQL*. O'Reilly.

Benjaafar, Saif, Sunderesh S. Heragu, and Shahrukh Irani. 2002. "Next Generation Factory Layouts: Research Challenges and Recent Progress." *Interfaces* 32 (6) (November): 58-76.

Benjaafar, Saifallah, and Mehdi Sheikhzadeh. 2000. "Design of flexible plant layouts." *IIE Transactions* 32 (4) (April): 309-322. doi:10.1080/07408170008963909.

Bergmann, Sören, and Steffen Strassburger. 2010. "Challenges for the Automatic Generation of Simulation Models for Production Systems." *Proceedings of the 2010 Summer Simulation Multiconference*: 545-549.

Bill Evjen, Billy Hollis, Rockford Lhotka, Rama Ramachandran Tim McCarthy, and Bill Sheldon Kent Sharkey. 2005. *Professional VB2005. Administrator*. Wiley Publishing, Inc.

Blumenau, J.C., and T. Kotz. 2004. Digital Factory vs. Lean Production Applied to Logistics- and Layout-Planning of Production Systems. P*aper presented at the International Seminar on Manufacturing Systems May 19-21, 2004*.

Bozer, Y. A., R. D. Meller, and S. J. Erlebacher. 1991. "An Improvement Type Layout Algorithm for Multiple Floor Facilities." *Technical Report* 11.

Bozer, Y.A., and R.D. Meller. 1994. "An improvement type layout algorithm for single and multiple-floor facilities." *Management Science* 40 (7): 918-932.

Buffa, E.S., and G.C. Armour. 1964. "Allocating facilities with CRAFT." *Harvard Business Review* 42: 136-158.

Burbidge, J. L. 1971. "Production flow analysis." *Production Engineer* (50): 139-152. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4913770.

Burgess, AG, I. Morgan, and TE Vollmann. 1993. "Cellular manufacturing: its impact on the total factory." *The International Journal of Production Research* 31 (9): 2059–2077.

Bzymek, Zbigniew M, Manuel Nunez, Mu Li, and Sean Powers. 2008. "Simulation of a Machining Sequence Using Delmia / Quest Software." *Computer-Aided Design and Applications* 5: 401-411.

Canen, Alberto G., and Geoff H. Williamson. 1996. "Facility layout overview: towards competitive advantage." *Facilities* 14 (10/11): 5-10.

Car, Zlatan, and Tonci Mikac. 2006. "Evolutionary approach for solving cell-formation problem in cell manufacturing." *Advanced Engineering Informatics* 20: 227-232.

Cardarelli, E., and P.M. Pelagagge. 1995. "Simulation tool for design and management optimization of automated interbay material handling and storage systems for large wafer fab." *Semiconductor Manufacturing* 8 (1): 44–49.

Carrie, AS. 1980. "Computer-aided layout planning—the way ahead." *International Journal of Production Research* 18 (3): 283–294.

Castillo, Ignacio, and Brett Peters. 2002. "Unit load and material-handling considerations in facility layout design." *International Journal of Production Research* 40 (13) (January): 2955-2989.

Chan, HM, and DA Milner. 1982. "Direct clustering algorithm for group formation in cellular manufacture." *Journal of Manufacturing Systems* 1 (1): 65–75.

Chee, Ailing. 2009. Facility layout improvement using systematic layout planning (SLP) and ARENA. Master thesis. Universiti Teknologi Malaysia.

Chen, Danfang. 2009. Software Tools for The Digital Factory – An Evaluation and Discussion. In *Proceedings of the 6th Cirp-Sponsored International Conference on Digital Enterprise Technology*, 803-812. Hong Kong: Springer-Verlag New York Inc.

Chen, Danfang, and Torsten Kjellberg. 2009. The Digital Factory and Digital Manufacturing – A Review and Discussion. In *CIRP Manufacturing Conference*. Vol. 1. Grenoble.

Cho, K., I. Moon, and W. Yun. 1996. "System analysis of a multi-product, small-lot-sized production by simulation: A Korean motor factory case." *Computers & industrial engineering* 30 (3): 347–356.

Christenson, K.R., and C.A. Dogan. 1995. "A simulation generator for dual-card kanban-controlled flow shops." *International journal of production research* 33 (9): 2615–2631.

Cormen, T.H. 2001. *Introduction to algorithms. New York*. The MIT press.

Curran, R., R. Collins, G. Poots, and T. Edgar. 2008. "Digital Lean Manufacture (DLM): A New Management Methodology for Production Operations Integration." *Life Cycle Management* 13: 551–571.

Davies, N.R. 1976. A modular interactive system for discrete event simulation modelling. In *Proceedings of the ninth international conference in systems simulation, Hawaii*, 296–299.

Dias, L.S., Pereira;G.B., and A.G. Rodrigues. 2007. "A Shortlist of the Most Popular Discrete Simulation Tools." *Simulation News Europe* 17: 33-36. doi:ISSN 0929-2268.

Dias, Luís M S, Guilherme A B Pereira, Pavel Vik, and José A Oliveira. 2011. Discrete Simulation Tools Ranking – A Commercial Software Packages Comparison Based on Popularity. In *Industrial Simulation Conference*, 5-11. Venice: Eurosis-ETI.

Dias, Luis, Pavel Vik, Guilherme Pereira, and José Oliveira., "Using 3D Simulation In Industry Internal Logistic Processes", in G. Janssens, K. Sörensen and C. Macharis (ed.), Decision Support in Supply Chain Management and Logistics, Cambridge Scholar Publishing, to be published in 2012.

Dilworth, J.B. 1992. *Operations Management: Design, Planning, and Control for Manufacturing and Services*. 2nd ed. McGraw Hill.

Donald, E.K. 1999. "The art of computer programming." *Sorting and Searching* 3.

Eneyo, E.S., and G.P. Pannirselvam. 1998. The use of simulation in facility layout design: a practical consulting experience. In *Proceedings of the 30th Winter simulation conference*, 1527–1532. IEEE Computer Society Press.

Erraguntla, M., P.C. Benjamin, and R.J. Mayer. 1994. An architecture of a knowledge-based simulation engine. In *Proceedings of the 26th Winter simulation conference*, 673–680. Society for Computer Simulation International.

Ferrari, E., A. Pareschi, A. Persona, and A. Regattieri. 2003. "Plant Layout Computerised Design : Logistic and Relayout Program (LRP)." *The International Journal of Advanced Manufacturing Technology* (April 2002): 917-922.

Finkelstein, Ellen. 2006. *AutoCAD 2006 and AutoCAD LT 2006 Bible*. Indianapolis: Wiley Publishing, Inc.

Foulds, L.R., and D.F. Robinson. 1976. "A strategy for solving the plant layout problem." *Operational Research Quarterly* 27: 845-855.

Francis, Richard, and John White. 1974. *Facilities Layout and Location*. New Jersey: Prentice Hall, Englewood Cliffs.

Fruggiero, Fabio, Via Ponte, and Don Melillo. 2006. Design and Optimization of a Facility Layout Problem. In *Proceedings of ICAD2006 4th International Conference on Axiomatic Design*, 31-37.

Fu, M.C., F.W. Glover, and J. April. 2005. Simulation optimization: a review, new developments, and applications. In *Proceedings of the Winter Simulation Conference*, 83-85. IEEE Computer Society Press.

Gani, John. 1957. "Problems in the probability theory of storage systems." *Journal of the Royal Statistical Society. Series B (Methodological)* 19 (2): 181–206.

Garces-Perez, J., D.A. Schoenefeld, and R.L. Wainwright. 1996. Solving facility layout problems using genetic programming. In *Proceedings of the First Annual Conference on Genetic Programming*, 182–190. MIT Press.

Garey, M.R., and D.S. Johnson. 1979. *Computers and intractability*. Vol. 174. New York: W.H. Freeman Press.

Gideon Halevi. 2001. *Handbook of production management methods*. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*. Vol. 216. Butterworth-Heinemann, January 1.

Ginsberg, A.S., H.M. Markowitz, and P.M. Oldfather. 1967. Programming by questionnaire. In *Proceeding of the AFIPS Conference*, 441–446. New York: ACM.

Goetschalckx, Marc P. 1992. *Computer aided design of industrial logistics systems*. Georgia Institute of Technology.

Grajo, E.S. 1995. Strategic layout planning and simulation for lean manufacturing a LayOPT tutorial. In *Proceedings of the 27th Winter simulation conference*, 510–514. IEEE Computer Society.

Green, RH, and L. Al-Hakim. 1985. "A heuristic for facilities layout planning." *Omega* 13 (5): 469–474.

Gupta, R.M. 1986. "Flexibility in layouts: a simulation approach." *Material Flow* 3: 243-250.

Gómez, Alberto, José Fernández Quesada, Isabel Parreño Fernández, and Nazario García Fernández. 2000. "The Use of Genetic Algorithms to Solve a Plant Layout Problem." *Assembly* (1975): 1-4.

Hamamoto, S. 1999. "Development and validation of genetic algorithm-based facility layout a case study in the pharmaceutical industry." *International Journal of Production Research* 37 (4): 749-768.

Hassan, M.M.D., and G.L. Hogg. 1991. "On constructing a block layout by graph theory." *International journal of production research* 29 (6): 1263–1278.

Havlík, Radek, Jan Vavruška, and František Koblasa. 2006. Optimalizace a inovace výrobních procesů (Optimization and innovation manufacturing processes). In *Proceeding of the MOPP Conference*, 69-74. Plzen.

Hellerstein, Joesph. 2007. "Architecture of a Database System." *Foundations and Trends in Databases* 1 (2): 141-259.

Heragu, Sunderesh S. 2006. *Facilities Design*. Second edi. Lincoln: iUniverse.

Hicks, P.E., and T.E. Lowan. 1976. "CRAFT-M for layout re-arrangement." *Industrial Engineering* 8 (5): 30-35.

Holland, John Henry. 1992. *Adaptation in natural and artificial systems: an introductory analysis with* *applications to biology, control, and artificial intelligence*. The MIT press.

Hosni, Y.A., G.E. Whitehouse, and T.S. Atkins. 1980. “MICRO-CRAFT Program Documentation.” *Institute of Industrial Engineers*.

Immer, J.R. 1950. *Layout planning techniques*. McGraw-Hill.

Irani, Shahrukh. 1999. *Handbook of Cellular Manufacturing Systems*. New York: John Wiley & Sons, Inc.

Jareš, David. 2008. Počítačová simulace transportního systému v podniku X (Computer simulation of transport system in the factory). Bachelor thesis. Technical University of Liberec.

Jennings, Roger. 2006. *Visual Basic 2005 Database Programming (2008). Solutions*.

Jensen, P., and Jonathan F. 2003. *Operations research: models and methods*. John Wiley and Sons.

Jiang, Z., Y. Huang, and J. Wang. 2010. Routing for the Milk-Run Pickup System in Automobile Parts Supply. In *Proceedings of the 6th CIRP-Sponsored International Conference on Digital Enterprise Technology*, 1267–1275. Springer.

Keller, Petr. 1996. Využití počítačů v projektování výrobních systémů (The Usage of Computers in the Producion System Design).

Kelley, James. 1961. “Critical Path Planning and Scheduling: Mathematical Basis.” *Operations Research* 9 (3).

Khoshnevisan, Mohammad. 2003. Optimal Plant Layout Design based on MASS Algorithm. In *Proceedings of the Sixth International Conference of Information Fusio*, 1365-1370. Cairns.

Kim, J.Y., and Y.D. Kim. 1985. “Graphic theoretic for unequal sized facility layout problems.” *International Journal of Management Science* 23: 391-401.

King, J.R. 1980. “Machine-component grouping in production flow analysis: an approach using a rank order clustering algorithm.” *International Journal of Production Research* 18 (2): 213–232.

Kjellberg, T, and Z Katz. 2005. The digital factory supporting changeability of manufacturing systems. In *Proceedings of the 38th CIRP International Seminar on Manufacturing Systems*. Florianópolis, Brazil.

Kleinrock, L. 1976. *Queueing Systems: Volume 2: Computer Applications*. Vol. 82. John Wiley & Sons.

Koopmans, T.C., and A Beckman. 1957. “Assignment problems and the location of economic activities.” *Econometrica: Journal of the Econometric* 25: 53-76.

Krieg, George. 2005. *Kanban-controlled manufacturing systems*. Springer-Verlag New York Inc.

Kulturel-Konak, S. 2007. "Approaches to uncertainties in facility layout problems: Perspectives at the beginning of the 21 st Century." *Journal of Intelligent Manufacturing* 18 (2): 273–284.

Kulturel-Konak, S., A.E. Smith, and B.A. Norman. 2004. "Layout optimization considering production uncertainty and routing flexibility." *International journal of production research* 42 (21): 4475–4493.

Kumara, S.R.T., RL Kashyap, and C.L. Moodie. 1988. "Application of expert systems and pattern recognition methodologies to facilities layout planning." *The International Journal Of Production Research* 26 (5): 905–930.

Kusiak, A., and W.S. Chow. 1987. "Efficient solving of the group technology problem." *Journal of manufacturing systems* 6 (2): 117–124.

Kwok, Martin. 1992. "Manufacturing System Design for the Industrial Engineer." *Industrial Engineering*: 35-39.

Law, A.M., and W.D. Kelton. 1991. *Simulation modeling and analysis*. Vol. 2. New York: McGraw-Hill New York.

Law, Averill M., and Michael G. Mccomas. 1997. Simulation of Manufacturing Systems. In *Manufacturing Systems*, 86-89.

Lee, R.C., and J.M Moore. 1967. "CORELAP-computerized relationship layout planning." *Journal of Industrial Engineering* (18).

Lee, Sanghoon, Hyunbo Cho, and Mooyoung Jung. 2000. "A conceptual framework for the generation of simulation models from process plans and resource configuration." *International Journal of Production Research* 38 (4): 811–828.

Leeder, E., Z. Ulrych, M. Bureš, Z. Černý, and J. Roubal. 2006. Digitální fabrika – Digital Factory. In *Proceedings of MOPP 2006 Conference*, 104-110.

Leeder, E., Z. Ulrych, M. Bureš, Z. Černý, and J. Roubal. 2007. Digitální fabrika – softwarové produkty pro oblast digitální fabriky (Digital factory - software products for DF). In *Proceedings of MOPP 2007 Conference*, 29-37.

Leung, J. 1992. "A graph-theoretic heuristic for designing loop-layout manufacturing systems." *European Journal of Operational* Research 57 (2).

Lucke, D., and C. Constantinescu. 2009. Context data model, the backbone of a smart factory. P*aper presented at 42nd CIRP Conference on Manufacturing Systems Sustainable Development of*

Manufacturing Systems. Grenoble.

Manlig, František. 1999. Aspekty a aplikační možnosti moderních simulačních systémů (The Aspects and Application Possibilites of Modern Simulation Systems). Technical University of Liberec.

Marasini, Ramesh, and N. Dawood. 2002. Simulation modeling and optimization of stockyard layouts for precast concrete products. In *Proceedings of the Winter Simulation Conference*, 2:1731–1736. IEEE Computer Society Press.

Marcoux, Nathalie, Diane Riopel, and A. Langevin. 2005. *Models and Methods for Facilities Layout Design from an Applicability to Real-World Perspective*. Ed. Springer. *Logistics systems: design and optimization*. New York: Springer.

Markt, P.L., and M.H. Mayer. 1997. WITNESS Simulation Software-A Flexible Suite of Simulation Tools. In *Proceedings of the Winter simulation conference*, 711–717. IEEE Computer Society Press.

Masurat, Thomas. 2009. White Paper on Results of Working Group " Open Digital Factory. ODF.

Mathewson, S.C. 1984. "The application of program generator software and its extensions to discrete event simulation modeling." *IIE transactions* 16 (1): 3–18.

McAuley, J. 1972. "Machine grouping for efficient production." *Production Engineer* 51 (2): 53–57.

McCormick, JR, P. J Scmwitzer, and T. W White. 1972. "Problem decomposition and data reorganization by a clustering technique." *Operations Research* 20: 993-1009.

McFedries, Paul. 2007. *VBA for the 2007 Microsoft Office System*. *Program*. Indianapolis: QUE.

Mecklenburg, Karsten. 2001. Seamless Integration of Layout and Simulation. In *Proceedings of the 2001 Winter Simulation Conference*, 1487. Citeseer.

Mertins, Kai, and Markus Rabe. 1995. Integration of Factory Simulation and CAD-Layout Planning. In *European Modelling and Simulation Symposium*, 77-81.

Meyers, F.E., and M.P. Stephens. 2005. *Manufacturing facilities design and material handling*. University of Michigan: Prentice Hall.

Moorthy, S. 2002. Integrating the CAD model with dynamic simulation: simulation data exchange. In *Proceedings of the Winter Simulation Conference*, 1:276–280. IEEE Computer Society Press.

Mujber, T.S., T. Szecsi, and M.S.J Hashmi. 2005. “A new hybrid dynamic modelling approach for process planning.” *Journal of Materials Processing Technology* 167 (August 2004): 22-32.

Murray, K.J., and S.V. Sheppard. 1988. “Knowledge-based simulation model specification.” *Simulation* 50 (3): 112.

Murugan, M., and V. Selladurai. 2005. “Manufacturing Cell Design with Reduction in Setup Time Through Genetic Algorithm.” *Journal of Theoretical and Applied Information Technology* 3: 76-97.

Muther, R. 1994. *Simplified systematic layout planning*. Third Edit. Marietta: Management & Industrial Research Publications.

Muther, R., Management, and Industrial Research Publications. 1973. *Systematic layout planning*. Management and Industrial Research Publications.

Muther, R., and J.D. Wheeler. 1994. *Simplified systematic layout planning*. Management and Industrial Research Publications.

Nylund, H., K. Salminen, and P. Andersson. 2008. “Digital Virtual Holons – An Approach to Digital Manufacturing Systems.” *Manufacturing Systems and Technologies for the New Frontier*: 103–106.

Otten, R.H.J.M. 1982. Automatic floorplan design. In *Proceedings of the 19th Design Automation Conference*, 261–267. IEEE Press.

Padillo, J.M., and D Meyersdorf. 1997. Incorporating manufacturing objectives into the semiconductor facility layout design process: a methodology and selected cases. In *Advanced Semiconductor Manufacturing Conference and Workshop*, 434 - 439.

Palekar, V.S., R. Batta, R.M. Bosch, and S. Elhence. 1992. “Modelling uncertainties in plant layout problems.” *European Journal of Operational Research* 63 (2).

Pandey, P. C., S. Janewithayapun, and M. a. a. Hasin. 2000. “An integrated system for capacity planning and facility layout.” *Production Planning & Control* 11 (8) (January): 742-753.

Parker, Frieda. 2008. “Space-Filling Curves.” *Math* 635.

Partovi, F. 1992. “An analytical hierarchy approach to facility layout.” *Computers & Industrial Engineering* 22 (74): 447-457.

Pegden, C.D., R.E. Shannon, and R.P. Sadowski. 1995. *Introduction to simulation using SIMAN*. Vol. 2. McGraw-Hill New York.

Peters, B.A., and T. Yang. 1997. “Integrated facility layout and material handling system design in semiconductor fabrication facilities.” *IEEE Transactions on Semiconductor Manufacturing* 10

(3): 360–369.

Powell, Gavin. 2006. *Beginning database design. Database.* Wiley.

Ranky, P.G., L.C. Morales, and R.J. Caudill. 2003. Lean disassembly line layout, and network simulation models. In *IEEE International Symposium on Electronics and the Environment,*, 19:36–41. IEEE.

Reed, Ruddel. 1967. *Plant location, layout, and maintenance.* Vol. 5. RD Irwin.

Reinhart, G. 2002. Information Becomes Knowledge – Automatic Building of Simulation Models. In *Proceedings of the 35th CIRP International Seminar*, 545-549. Seoul, Korea.

Robinson, S., RJ Brooks, K. Kotiadis, and D.J. van der Zee. 2010. *Conceptual modelling for discrete-event simulation.* CRC Press.

Rooks, T. 2009. Rechnergestützte Simulationsmodell- generierung zur dynamischen Absicherung der Montagelogistikplanung bei der Fahrzeugneutyp- planung im Rahmen der Digitalen Fabrik (Computer supported simulation model generation for dynamic validation of assembly logist. Doctoral thesis.TU Clausthal-Zellerfeld.

Rosenblatt, M.J. 1979. "The facilities layout problem: a multi-goal approach." *International Journal of Production Research* 17 (4): 323–332.

Sahni, Sartaj, and Teofilo Gonzalez. 1976. "P-Complete Approximation Problems." *Journal of the Association for Computing Machinery* 23 (3): 555-565.

Satoh, I. 2008. "A Formal Approach for Milk-Run Transport Logistics." *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences* 91 (11): 3261–3268.

Schroer, B.J., and F.T. Tseng. 1989. "An intelligent assistant for manufacturing system simulation." *International journal of production research* 27 (10): 1665–1683.

Scriabin, M, and R.C Vergin. 1975. "Comparison of computer algorithms and visual based methods for plant layout." *Management Science* 22 (2).

Seehof, J.M., and W.O. Evans. 1967. "Automated layout design program." *Journal of Industrial Engineering* 18 (12): 690–695.

Sekine, K. 1992. *One-Piece Flow: Cell Design for Transforming the Production Process.* Productivity Press.

Seppanen, Marvin, S. 2000. Developing Industrial Strength Simulation Models Using Visual Basic for Applications (VBA). In *Proceedings of the 2000 Winter Simulation Conference*, 77-82. Citeseer.

Ševčík, L. 2002. *Úvod do programování v CAD: Visual LISP, Visual Basic, JAVA, Pro/PROGRAM pro AutoCAD, Inventor, Pro/Engineer (Introduction to CAD programming)*. Liberec: Technická univerzita v Liberci.

Shih, L.C., T. Enkawa, and K. Itoh. 1992. "An AI-search technique-based layout planning method." *The International Journal Of Production Research* 30 (12): 2839–2855.

Sims R. 1990. "MH problems are business problems." *Industrial Engineering* (May).

Singh, S. P., and R. R. K. Sharma. 2005. "A review of different approaches to the facility layout problems." *The International Journal of Advanced Manufacturing Technology* 30 (5-6) (November 12): 425-433.

Sly, David. 1993. AutoCAD-based industrial layout planning and material flow analysis in FactoryFLOW and PLAN. In *Proceedings of the 25th Winter simulation conference*, 281–284. ACM.

Sly, David. 1995. Plant Design for Efficiency Using Autocad and FactoryFlow. In *Proceedings of the Winter Simulation Conference*, 437 - 444. Arlington , USA: IEEE Computer Society Press.

Sly, David. 1997. Before dynamic simulation: systematic layout design from scratch. In *Proceedings of the 29th conference on Winter simulation*, 645:645–648. IEEE Computer Society.

Sly, David, and S. Moorthy. 2001. Simulation data exchange (SDX) implementation and use. In *Proceedings of the Winter Simulation Conference*, 2:1473–1477. IEEE Computer Society Press.

Smutkupt, Uttapol, and Sakapoj Wimonkasame. 2009. "Plant Layout Design with Simulation." *Proceedings of the International MultiConference of Engineers and Computer Scientists* 2 (1): 18–20.

Sniedovich, M. 2010. *Dynamic programming: foundations and principles*. Second edi. Vol. 273. CRC Press.

Sniedovich, M. 2006. "Dijkstra's algorithm revisited: the dynamic programming connexion." *Control and cybernetics* 35 (3): 599.

Son, Y.J., and R.A. Wysk. 2001. "Automatic simulation model generation for simulation-based, real-time shop floor control." *Computers in Industry* 45 (3): 291–308.

Stephens, Rod. 2005. *Visual Basic 2005 Programmer's Reference*. Indianapolis: Wiley Publishing, Inc.

Sutphin, Joe. 2006. *AutoCAD 2006 VBA: A Programmer's Reference*. New York: Apress.

Tam, K.Y., and S.G. Li. 1991. "A hierarchical approach to the facility layout problem." *International Journal of Production Research* 29 (1): 165–184.

Tanchoco, J.M.A. 1994. *Material flow systems in manufacturing*. Cambridge: Springer.

Tang, C., and L.L. Abdel-Malek. 1996. "A framework for hierarchical interactive generation of cellular layout." *International journal of production research* 34 (8): 2133–2162.

Taylor, G. 2009. *Introduction to logistics engineering*. Boca Raton: CRC Press.

Tompkins, James, A., A. White, John, A Bozer, Yavuz, and J.M.A. Tanchoco. 2010. *Facilities planning*. Fourth edi. John Wiley and Sons.

Urban, T.L. 1987. "A multiple criteria model for the facilities layout problem." *International Journal of Production Research* 25 (12): 1805–1812.

Vik, Pavel. 2007. The Example of Usage of Computer Simulation. In *Proceeding of the MOPP Conference (Mopp 2007)*, 214-218. Plzeň: University of West Bohemia.

Vik, Pavel, Luis Dias, and José Oliveira. 2008a. Using of 3D Simulation in the Industry. P*aper presented at the conference 2nd. International Conference of Technology Knowledge and Information (ICTKI)*. Ústí nad Labem.

Vik, Pavel, Luis Dias, José Oliveira, and Guilherme Pereira. 2008b. Using 3D Simulation in an Internal Logistic Process. In *Proceedings of 15th European Concurrent Engineering Conference (ECEC2008)*, 116-120. Porto: Porto April 2008. EUROSIS-ETI Publication.

Vik, Pavel, and David Jareš. 2008. Využití počítačové simulace v interní logistice (Usage of computer simulation in the internal logistics). In *Sborník příspěvků 11. ročníku mezinárodní konferenc*, 81-85. HUMUSOFT s.r.o. & VUT Brno.

Vik, Pavel, Luis Dias, Guilherme Pereira, José Oliveira, and R. Abreu. 2010a. Using Simio for the Specification of an Integrated Automated Weighing Solution in a Cement Plant. In *Proceedings of the 2010 Winter Simulation Conference*, 1534-1546. Baltimore, USA.

Vik, Pavel, Luis Dias, Guilherme Pereira, José Oliveira, and Ricardo Abreu. 2010b. Using SIMIO in the Design of Integrated Automated Solutions For Cement Plants. In *Workshop on Applied Modelling and Simulation*. Rio de Janeiro, Brasil: Universidade Federal do Rio.

Vik, Pavel, Luis Dias, Guilherme Pereira, and José Oliveira. 2010c. Improving Production and Internal Logistics Systems - An Integrated Approach Using CAD and Simulation. In *Proceedings of the 3rd International Conference on Information Systems, Logistics and Supply Chain*. Casablanca (Morocco).

Vik, Pavel, Luís Dias, Guilherme Pereira, and José Oliveira. 2010d. Automatic Generation of Computer Models through the Integration of Production Systems Design Software Tools. P*aper presented at the conference 3th International Conference on Multidisciplinary Design Optimization and*

Applications. Paris, France.

Vilarinho, P., and R. Guimaraes. 2003. "A Facility Layout Design Support System." *Investigacao Operacional* 23 (1): 145–161.

Věchet, Vladimír. 1982. *Technologické projekty (Technological projects)*. Liberec: Technical University of Liberec.

Weinberg, P. N., J. R. Groff, J. Groff, and A .J. Oppel. 2009. *SQL, the complete reference*. McGraw-Hill.

Wemmerliiv, U., and N. L. Heyer. 1989. "Cellular manufacturing in the U.S . industry: a survey of users." *International Journal of Production Research* 27 (9).

Weng, L. 1999. Efficient and flexible algorithm for plant layout generation. West Virginia University.

Westkaemper, E., L. Jendoubi, M. Eissele, and T. Ertl. 2005. Smart Factory-Bridging the gap between digital planning and reality. In *proceedings of the 38th CIRP International Seminar on Manufacturing Systems*, 44–44.

Whitman, L.E., M. Jorgensen, K. Hathiyari, and D. Malzahn. 2004. Virtual reality: its usefulness for ergonomic analysis. In *Proceedings of the 36th conference Winter simulation*, 1740–1745. Winter Simulation Conference.

Williams, E.J. 1998. Quantitative Process Layout Design Supported by Discrete Simulation. In *Industrial Engineering Research '98 Conference Proceedings*, 1-7. Citeseer.

Wilsten, Pinto, and E. Shayan. 2007. "Layout Design of a Furniture Production Line Using Formal Methods." *Journal of Industrial and Systems Engineering* 1 (1): 81-96.

Wiyaratn, W, and A Watanapa. 2010. "Improvement Plant Layout Using Systematic Layout Planning (SLP) for Increased Productivity." *Engineering and Technology* (72): 373-377.

Worn, H., D. Frey, and J. Keitel. 2002. Digital factory-planning and running enterprises of the future. In *Industrial Electronics Society, 2000. IECON 2000. 26th Annual Confference of the IEEE*, 2:1286–1291. IEEE.

Wu, Y., and E. Appleton. 2002. "The optimization of block layout and aisle structure by a genetic algorithm." *Computers and Industrial Engineering* 41 (4): 371–387.

Wurdig, T., and R. Wacker. 2008. "Generische Simulationslösung für Fördertechnik (Generic Simulation for Conveyor Systems)." *Advances in Simulation for Production and Logistics Applications*: 11–20.

Yang, M., and J. Yang. 2008. "Machine-part cell formation in group technology using a modified ART1 method." *European Journal of Operational Research* 188 (1): 140-152.

Yuan, Y., C.A. Dogan, and G.L. Viegelahn. 1993. "A flexible simulation model generator." *Computers & industrial engineering* 24 (2): 165–175.

Zelenka, Antonín, and Mirko Král. 1995. *Projektování výrobních systémů (Design of production system)*. Praha: ČVUT.

Zülch, G., and S Stowasser. 2005. "The Digital Factory: An instrument of the present and future, Computer in industry." *Computer in industry* 56 (1): 323-324.

Appendix 1 Installation comments

In this section, some comments are described about installation of tools. All "dll" or "ocr" libraries are included in standard Office package.

A.1.1 AutoCAD reference

Following controls panels and external reference must be loaded to the VBA AutoCAD:

Available Controls (AutoCAD VBA OCX)

- AutoCAD DwgThumbnail Control
- Microsoft Forms 2.0 ComboBox
- Microsoft Forms 2.0 Command Button
- Microsoft Forms 2.0 Frame
- Microsoft Forms 2.0 CheckBox
- Microsoft Forms 2.0 Image
- Microsoft Forms 2.0 Label
- Microsoft Forms 2.0 ListBox
- Microsoft Forms 2.0 MultiPage
- Microsoft Forms 2.0 OptionButton
- Microsoft Forms 2.0 ScrollBar
- Microsoft Forms 2.0 Spin Button
- Microsoft Forms 2.0 TabStrip
- Microsoft Forms 2.0 TextBox
- Microsoft Forms 2.0 Toggle Button
- Microsoft TreeView Control, vs 6

Available References

- Visual Basic For Applications
- OLE Automation
- Microsoft DAO 3.6 Object Library
- Microsoft Access 14.0 Object Library
- Microsoft Forms 2.0 Object Library
- AutoCAD 2011 Type Library
- Microsoft Office 14.0 Object Library
- AutoCAD Focus Control for VBA Type Library
- Microsoft Windows Common Controls 6.0 (SP6)
- Microsoft Common Dialog Control 6.0 (SP3)
- Microsoft Windows Common Controls 5.0 (SP2)
- AutoCAD DwgThumbnail Control module
- Microsoft Internet Controls
- Microsoft ActiveX Data Objects 6.0 Library
- Microsoft Scripting Runtime

A.1.2 MS Access reference

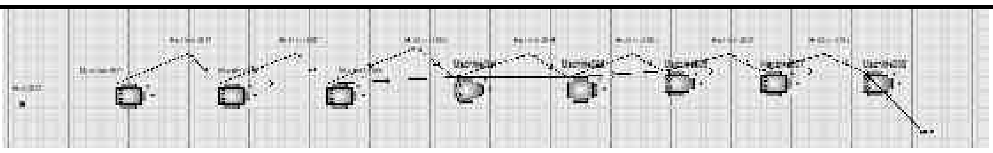
- Visual Basic For Appkcabons
- Microsoft Access 14.0 Object Library
- OLE Automation
- Microsoft DAO 3.6 Object Library
- Microsoft Common Dialog Control 6.0 (SP3)
- Microsoft Forms 2.0 Object Library
- **Microsoft ActiveX Data Objects 2.8 Library**
- **Microsoft Windows Common Controls 5.0 (SP2)**
- **Microsoft Windows Common Controls 6.0 (SP6)**
- **Microsoft Office 14.0 Object Library**

Appendix 2 Importance of buffers in the system

Buffers are very important elements in the networked system to cover uncertainties or non-synchronized systems as buffers can rapidly improve throughput of systems, avoid job waiting or block state.

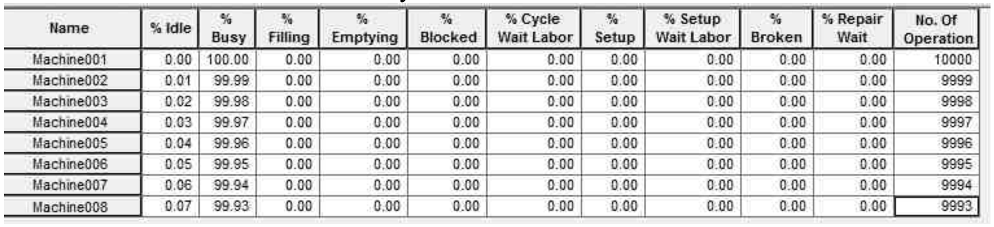
In Table 35, simple simulation model and results of the simulation experiments are shown. There are 8 machines placed in the production line (serialized line) and parts go through each machine consecutively. This model is based on the work of Williams (1998), who described the need of buffers in the system.

Table 35 – Importance of buffers in the system



Simulation model made in WITNESS, with icons representing machines and buffers. Objects are connected by flows meaning control logic and material flows

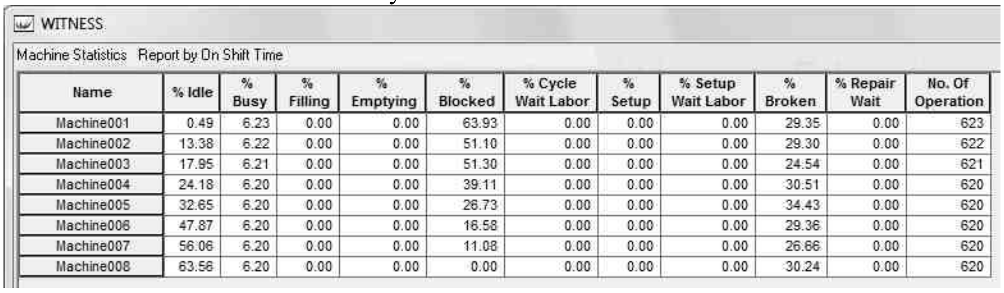
System without fails:



Name	% Idle	% Busy	% Filling	% Emptying	% Blocked	% Cycle Wait Labor	% Setup	% Setup Wait Labor	% Broken	% Repair Wait	No. Of Operation
Machine001	0.00	100.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	10000
Machine002	0.01	99.99	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	9999
Machine003	0.02	99.98	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	9998
Machine004	0.03	99.97	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	9997
Machine005	0.04	99.96	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	9996
Machine006	0.05	99.95	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	9995
Machine007	0.06	99.94	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	9994
Machine008	0.07	99.93	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	9993

Total throughput is 9993 parts per 10 000 minutes

System without buffers:



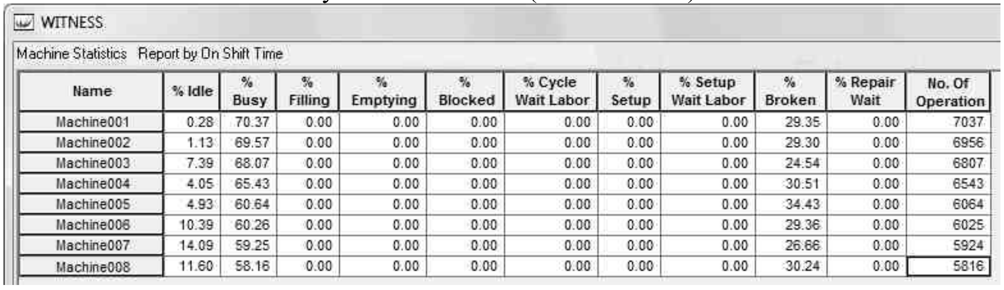
WITNESS

Machine Statistics Report by On Shift Time

Name	% Idle	% Busy	% Filling	% Emptying	% Blocked	% Cycle Wait Labor	% Setup	% Setup Wait Labor	% Broken	% Repair Wait	No. Of Operation
Machine001	0.49	6.23	0.00	0.00	63.93	0.00	0.00	0.00	29.35	0.00	623
Machine002	13.38	6.22	0.00	0.00	51.10	0.00	0.00	0.00	29.30	0.00	622
Machine003	17.95	6.21	0.00	0.00	51.30	0.00	0.00	0.00	24.54	0.00	621
Machine004	24.18	6.20	0.00	0.00	39.11	0.00	0.00	0.00	30.51	0.00	620
Machine005	32.65	6.20	0.00	0.00	26.73	0.00	0.00	0.00	34.43	0.00	620
Machine006	47.87	6.20	0.00	0.00	16.58	0.00	0.00	0.00	29.36	0.00	620
Machine007	56.06	6.20	0.00	0.00	11.08	0.00	0.00	0.00	26.66	0.00	620
Machine008	63.56	6.20	0.00	0.00	0.00	0.00	0.00	0.00	30.24	0.00	620

Total throughput is 620 parts per 10 000 minutes

System with buffers (unlimited size)



WITNESS

Machine Statistics Report by On Shift Time

Name	% Idle	% Busy	% Filling	% Emptying	% Blocked	% Cycle Wait Labor	% Setup	% Setup Wait Labor	% Broken	% Repair Wait	No. Of Operation
Machine001	0.28	70.37	0.00	0.00	0.00	0.00	0.00	0.00	29.35	0.00	7037
Machine002	1.13	69.57	0.00	0.00	0.00	0.00	0.00	0.00	29.30	0.00	6956
Machine003	7.39	68.07	0.00	0.00	0.00	0.00	0.00	0.00	24.54	0.00	6807
Machine004	4.05	65.43	0.00	0.00	0.00	0.00	0.00	0.00	30.51	0.00	6543
Machine005	4.93	60.64	0.00	0.00	0.00	0.00	0.00	0.00	34.43	0.00	6064
Machine006	10.39	60.26	0.00	0.00	0.00	0.00	0.00	0.00	29.36	0.00	6025
Machine007	14.09	59.25	0.00	0.00	0.00	0.00	0.00	0.00	26.66	0.00	5924
Machine008	11.60	58.16	0.00	0.00	0.00	0.00	0.00	0.00	30.24	0.00	5816

Total throughput is 5816 parts per 10 000 minutes

The operation time of each machine is one minute (constant value). These machines have breakdowns defined by time to repair by gamma distribution function (shape parameter $\alpha = 1.67$ and scale parameter $\beta = 18$) and time between failures (exponential distribution function (parameter mean=100).

In the first case, the system doesn´t involve buffers between machines. In the second case, the model contains buffers where parts are stored if necessary.

From the results (Table 35), machines work around up 6% only, and they are blocked in even from 64%. In the second case (system with buffers), machines work on around 60%. The system productivity (% rate between total throughput of current system and throughput of system without failures) is increased from 6% (=662/9993) to 58%.

Appendix 3 Programme and algorithms codes

A.3.1 Introduction

Programme codes involved in IDS contains around 800 pages. The most important parts of the code are mentioned as pseudo codes or in full version in the body of the thesis. It is not feasible to insert full codes of all functions and algorithms to the thesis, therefore there are available in the internet web page:

http://dl.dropbox.com/u/1419597/_additional_materials_thesis.zip

or also on the project web page: **www.ids.bluefile.cz**

Following sections contains lists of files with full paths and functions. Complete codes and structure are involved in the basic files ("database.mdb", "CAD_IDS.dvb" and "simulation_model_IDS.mod").

A.3.2 Database functions

Files concerning database files are involved in Table 36.

Table 36 – Database files

File name and path	Comments
_database\database_IDS.zip	"database.mdb" file consists
Module_Cluster.txt	Cluster analysis – ROC and DCA algorithms (**IDS. PRODUCTION ANALYSE. CLUSTER_ANALYSIS**)
Module_Simple_department.txt	Function for data preparing (**IDS. DETERMINISTIC_MODELS**)
Module_WITNESS_design.txt	Functions of WITNESS generation models (**IDS.GENERATION_SIM MODELS**)

A.3.3 AutoCAD functions

Basic functions for the work with blocks are inserting, adding attributes, loading information from block instances and deleting blocks.

A.3.3.1 Loading information from block instances

This procedure searches for all objects in AutoCAD. If the object is a block with attributes (e.g. "PART_NAME"), information is saved in DB (see Figure 165).

```
Dim elem As AcadEntity
Dim arrBlockAtr As Variant

For Each elem In ThisDrawing.ModelSpace
'Check if object is block:
If StrComp(elem.EntityName, "AcDbBlockReference", 1) = 0 Then
```

```
'block containing attributes:

      If elem.HasAttributes Then
      arrBlockAtr = elem.GetAttributes

'basic information about objects:

      insert_point = elem.InsertionPoint
      x_coordinates = insert_point(0); y_coordinates = insert_point(1);
      z_coordinates = insert_point(2)
      x_zoom = elem.XScaleFactor; y_zoom = elem.YScaleFactor; z_zoom =
      elem.ZScaleFactor

      angle_rotation = elem.Rotation
      reference_name = elem.Name
      block_layer = elem.layer

            For i_i = LBound(arrBlockAtr) To UBound(arrBlockAtr)

                  If arrBlockAtr(i_i).TagString = "PART_NAME" Then
                  atrPart_name = arrBlockAtr(i_i).TextString
            End If

            If arrBlockAtr(i_i).TagString = "NAME_BUFFER" Then
            atrName_buffer = arrBlockAtr(i_i).TextString
            End If

      Next 'i_i

insert_sql = "INSERT INTO Tbl_Process_Blocks_coordinates
(Reference_name,block_layer, x_coordinates, y_coordinates,…Buffer_name…)
VALUES(" & Chr(39) & reference_name & Chr(39) & "," & Chr(39) & block_layer
& Chr(39) & ","… atrName_buffer…)
DoCmd.RunSQL (insert_sql)

End If
End If
Next
```

Figure 165 - Procedure: Loading information from block instances

A.3.3.2 Deleting blocks procedure

It is possible to delete concrete block instances with the previous code or by deleting all objects in a given layer (see Figure 166).

```
For Each elem In ThisDrawing.ModelSpace
      If elem.layer = "Layout_process" Then
      elem.Delete
      End If
Next
```

Figure 166 - Procedure: Deleting blocks procedure

Dynamic block

Dynamic blocks are used in IDS for generating resources or storages with rectangular shape and given dimensions. These dimensions are parameters of the block (see Figure 167).

```
Set xref = ThisDrawing.ModelSpace.InsertBlock(pointStart, strHelper, 1, 1,
1, 0)

If xref.IsDynamicBlock = True Then
      Props = xref.GetDynamicBlockProperties

      For k_k = 0 To UBound(Props)
            If Props(k_k).PropertyName = "Distance_x" Then
            Props(k_k).Value = arrMachines_Dim_x(i_i)
            End If
            If Props(k_k).PropertyName = "Distance_y" Then
            Props(k_k).Value = arrMachines_Dim_y(i_i)
            End If
      Next 'j_j

End If
```

Figure 167 - Procedure: Dynamic block

Files concerning AutoCAD files are involved in Table 37.

Table 37 – AutoCAD files

File name and path	Comments
CAD_IDS.dvb	Complete AutoCAD VBA project with forms, codes etc.
CAD_blocks_library.zip	Drawing blocks in the library structure
Module_CRAFT.txt	Function for CRAFT method implementation **IDS. FACTORY.CRAFT** **IDS. FACTORY.MANUAL**
Module_AutoCAD_function.txt	General function for work in AutoCAD, placement functions, transportation functions **IDS.FACILITY.FACILITY PLACEMENT** **IDS.FACILITY.BUFFER SOLUTION** **IDS.FACILITY.WITNESS** **IDS.FACILITY.TRANSPORT FUNCTION**
Module_Declarations_variable.txt	Declaration of variables
Module_CAD_Library_Browser.txt	Function for browsing in the disc **IDS.FACILITY.CAD LIBRARY** **IDS.FACILITY.BLOCKS**
Module_CORELAP.txt	Function for CORELAP method implementation **IDS.FACTORY.CORELAP**
Module_GRAPHIC.txt	Function for Graphic block layout method implementation **IDS. FACTORY. GRAPH_METHOD**
Module_Material_Flows.txt	Function for generation of polylines and work with material flows **IDS.FACILITY. MATERIAL FLOWS**
Module_ROC.txt	Function for clusters placement **IDS.FACILITY. FACILITY PLACEMENT**
Module_Shortest_path.txt	Function including Dijkstra algorithm and its implementation **IDS.FACILITY. MATERIAL FLOWS**
Module_Simple_department.txt	Function for deterministic layout **IDS.FACILITY.SIMPLE_MODELS**

A.3.3.3 Simulation functions

The following table (Table 38) summarises all functions. The complete code can be found in Appendix (Section A.3.3.3). Table 39 contains summarised phases of initialisation. Files concerning simulation files are involved in.

Table 38 – Functions used in the simulation

Function	Comments
Functions for objects details definition and initialisation	
Initialisation()	Set of functions for initialisation of model, run automatically
function_Detail()	Definition of specific properties of objects
fce_Load_status()	Initialisation of objects from external file
Functions for internal logic of resource and buffers	
fce_Machine_AI_last, fce_Machine_AI_inc, fce_Machine_AI_first, fce_Machine_AF, fce_Machine_AO, fce_Machine_OR_bu, fce_Machine_OR_li	Set of functions defined resource internal logic control
Functions for external logic	
function _Priority, function _Priority_previous	Functions for searching operation for given resource, bases on parameters
Functions for operation	
fce_Timing_buffer	Definition of minimum time delay of entity in the buffer
fce_Part_wait	Properties of operation, allow waiting for all of the inputting entities
fce_Unpack_previous fce_Unpack_next	Unpacking operations definition
fce_Blocation	Function defined of resources blocked states
fce_Setuping	Function for setup time definition on the resources
Functions for transportation	
function_Transport_time	Calculation of transportation time based on start and final station
function_Transport_time_tp	Calculation of transportation time based on the current position of transporter and start station
function _MR_load_cy, function _Change_io function _Milk_run,	Function for milk-run definition
Functions for general issues	
function_Position_object	Placement of simulation objects based on CAD drawing
fce_Array	Arrays sizing and loading data from the DB
function _Find_in_buffer	Finding of specific entity in the buffer based on the attributes
function_Clean_files	Deleting previous data
function_Write_Storage_Data RecordStringValue (Dyn_array_Buffer_name,ELEMENT)	Writing data about buffer behaviour into external file, recorded by a set of functions *RecordStringValue*, values: element name, time and number of parts
fce_Finish_process_statistics fce_Reset_Statistics()	Finishing resources statistics and resetting statistics data
fce_Profiles	Definition of profiles

Table 39 – Initialisation function phases

Phase of initialisation	**Description**
database_connection	Establishing connection with the DB
Setup of initialisation	Declaration of variables, function *fce_clean_files* () for deleting previous data
function_Arrays ()	Loading data from DB into arrays in WITNESS, see example below
ImportState("file.sta") fce_Load_status()	Preparing model based on data as entities in buffers, resources states (busy, idle, break, etc.) with WIP, statistics of objects, etc., see Section A.4.3
function_Detail ()	Specification of objects details, see example below
RULE ON INPUT	Resource: Rules on input [see Section 4.4.3 a)], see also example below
ACTION ON INPUT	Resource: Action on Input [see Section 4.4.3 b)]
ACTION ON START	Resource: Action on Start [see Section 4.4.3 c)]
ACTION ON FINISH	Resource: Action on Finish [see Section 4.4.3 d)]
RULE ON OUTPUT	Resource: Rule on Output [see Section 4.4.3 e)]
ACTION ON OUTPUT	Resource: Action on Output [see Section 4.4.3 f)]
BUFFER - ACTION ON INPUT	Buffer: Action on Input [see Section 4.4.4 a)]
BUFFER - ACTION ON OUTPUT	Buffer: Action on Output [see Section 4.4.4 c)]
ACTION PREBUFFER	Buffer: Action on Input, Output [see Section 4.4.4]
OUTPUTSHIP	Resource: complete details [see Section 4.4.6 b)]
fce_Milk_run ()	Transportation: milk-run
Save model after whole initialisation	Save model after whole initialisation

Table 40 – Simulation files

File name and path	Comments
simulation_model_IDS.zip	"simulation_model_IDS.mod" Default WITNESS simulation model involving complete structure
initialise_action.wcl	Initialisation set of actions
_data_processing	
fce_Detail.txt	Definition of objects details
fce_Load_status.txt	Function for load status (warm-up time)
fce_Dynamic_Array.txt	Function for saving historical data of buffers
fce_Array.txt	Function for loading data from DB to simulation model
fce_Write_Storage_Data.txt	Function for export historical data of buffers
fce_Clean_files.txt	Function for clean all external files
fce_Reset_Statistics.txt	Function for reset of statistics (warm-up time)
fce_Finish_process_statistics.txt	Function for finishing operation and resource statistics
_external_logic_control	
funkce_Priority_next.txt	Function for selecting operation for resource
funkce_Priority_previous.txt	Function for selecting operation for resource
_internal_logic_control	
fce_Machine_AO.txt	Function for Action on Output (resource)
fce_Machine_AF.txt	Function for Action on Finish (resource)
fce_Machine_AI_inc.txt	Function for Action on Input (resource)
fce_Machine_AI_last.txt	Function for Action on Input (resource)
\fce_Machine_OR_li.txt	Function for Output rule – buffer limitation (resource)
fce_Machine_OR_bu.txt	Function for Output rule – buffer limitation (resource)
fce_Machine_AI_first.txt	Function for Action on Input (resource)
_operation_functions	
fce_Blocation.txt	Function for "Waiting free space in output buffer"
fce_Timing_buffer.txt	Function for definition of operation time (buffers)
fce_Position_object.txt	Function for placing simulation objects based on CAD layout
fce_Part_wait.txt	Function for "part waiting" operation property
fce_Unpack_previous.txt	Function for unpacking operation
fce_Unpack_next.txt	Function for unpacking operation
fce_Find_in_buffer.txt	Function for searching entities in the buffer
fce_Timing.txt	Function for definition of operation time
fce_Setuping.txt	Function for definition of setup time
_transport_functions	
fce_Transport_time.txt	Function for calculation of transportation time of main transporter
fce_Transport_time_tp.txt	Function for calculation of transportation time of helper transporter
fce_Milk_run.txt	Function for milk-run definition
fce_Milk_Run_ow.txt	Function for milk-run definition
fce_Change_io.txt	Function for milk-run – changing input/output amount
fce_MR_load_cy.txt	Function for milk-run – load cycle
_others	
Module_JIT.bas (JIT_order.xlsm)	JIT order generator
Module_Calculation.bas	Analysis of number unique entities in defined time period
STA_generator.xls	Function for STA files generator (MS Excel)

Appendix 4 Description of IDS functionalities and applications

This chapter is focused on the explanation of the work in the IDS system by the use of forms in integrated tools. The functions involved in forms are explained in simple models and examples for an easier understanding. In the next sections, the following structure of presented forms and functions is used:

Property of table:

IDS.DB.OPERATIONSETUP.LISTOFOPERATION.*kind_of_operation*=”production”

This structure means: in the **IDS** database (**DB**), in the form **OPERATIONSETUP**, in the included table **LISTOFOPERATION**, the property in the column ***kind_of_operation*** has to be setup to value “production”.

Run function:

IDS.WITNESS.USERACTION.*function_Write_Storage_Data()*

It means that in WITNESS, in User Action menu *function_Write_Storage_Data()* is launched by the user.

IDS.CAD.FACILITY.MATERIAL_FLOWS.Saving limited flows()

In this case, in the form **MATERIAL_FLOWS**, a function “*Saving limited flows()*” is launched.

Setup of properties:

IDS.CAD.Facility.Material_Flows.*Transparency* = true/false

Parameter “Transparency” can be setup as true or false.

IDS.WITNESS.Initialisation.*boo_Dyn_write* = 1

Parameter “boo_Dyn_write” must be setup as 1.

Properties are in italic font and functions are in italic font and underlined.

A.4.1 Library of CAD drawings

IDS provides a library of drawing blocks that can be used in the layout. As mentioned in Section 4.5.2, each block must contain specific information in its attributes to distinguish itself uniquely (attribute “name”). Besides input and output, blocks representing resources and buffers have points for definition of material flows (see Figure 86 and Figure 168). Working with blocks drawing has the following parts:

- Preparing blocks
- Inserting into the library
- Using in layout drawing

a) <u>Preparing blocks</u>

The block must be prepared before it is inserted into the CAD library. The correct scale, position and units in millimetres must be established. Each block must contain specific attributes for the unique determination of the object in the layout as e.g. *NAME_PROCESS* (see full list in Table 20.

b) <u>Inserting into the library as a new item</u>

Prepared drawing blocks can be inserted into the IDS library through **IDS.CAD.FACILITY.CAD_LIBRARY** form. Path of drawing file and block name is specified. Then this file is copied to the selected folder (see Figure 168).

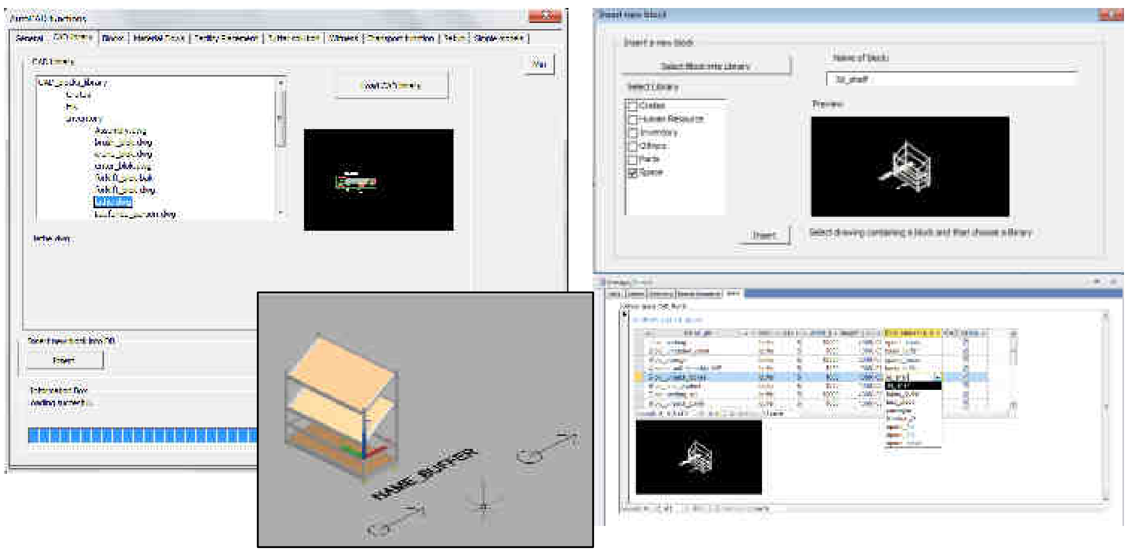


Figure 168 - Block preparation, CAD library

c) <u>Using in layout drawing</u>

In the form **IDS.DB.RESOURCES**, it is possible select drawing block for resource, buffer or entity in property *"Block reference"*. Functions automatically insert blocks into layouts and fill attributes information.

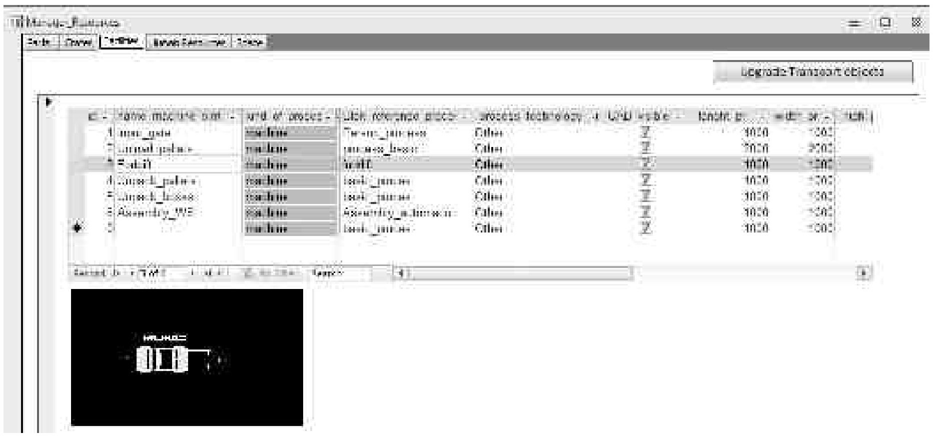
A.4.2 System elements properties

A.4.2.1 Resources

Objects *Resources* were already introduced in the previous sections 4.2.5 and 4.4.3. The following sections describe important functions and their use in practical solving examples.

Resource properties

A resource object can be a production machine, a transporter or other equipment. Form in Figure 169 (**IDS.DB.RESOURCES.FACILITIES**) is the main input form for managing resources. Resources can be configured through this form as: *unique ID number* and *name*, *kind of resource block reference process*, *kind of technology*, *dimensions,* etc. Property "*kind_of_resource*" determines basic functions of the resource (machine, crane, milk-run, and quantity function). Drawing block is selected in the field "*Block reference process*". "*Kind_of_technology*" defines the kind of production technology; it is used in the job-shop layouts example in Section A.4.9.



id	Name machine unit	Kind of process	Block reference process	process technology	CAD visible	length	width
1	[illegible]_gate	machine	[illegible]_process	Other	☒	1000	1000
2	[illegible]_palace	machine	process_base	Other	☒	1000	1000
3	Forklift	machine	forklift	Other	☒	1000	1000
4	Unpack_pallets	machine	task_process	Other	☒	1000	1000
5	Unpack_boxes	machine	task_process	Other	☒	1000	1000
6	Assembly_WS	machine	Assembly_automation	Other	☒	1000	1000
0		machine	task_process	Other	☒	1000	1000

Figure 169 - Resource input form

Resources states during simulation

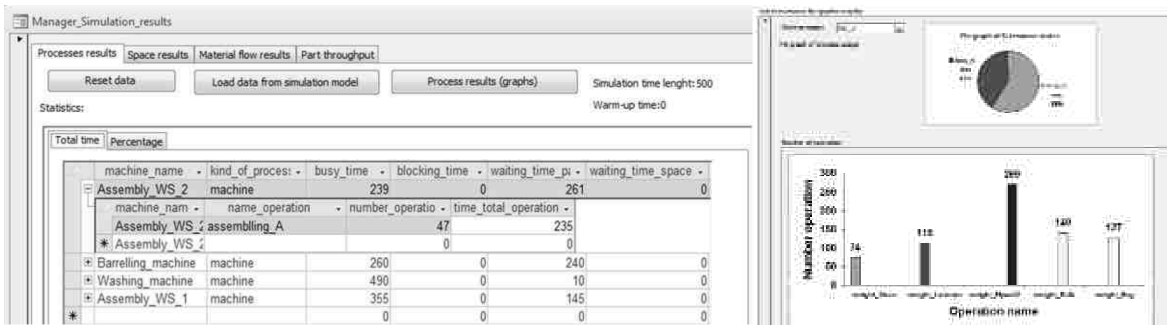
One of the very important tasks, performed by simulation, is the analysis system resources statistics. Values as the percentage of working time period, either in idle or at break time periods, are very helpful for production planning and layout design.

After running the simulation experiments, function **IDS.WITNESS.USER ACTIONS.*<u>fce Finish statistics()</u>*** finishes the statistics of resources. Then, the function **IDS.DB.SIMULATION_RESULTS. *<u>Load data from simulation()</u>*** loads these values to DB (see Figure 170). The WITNESS default statistics' report is spread by the operation statistics as the number of operation processed on each machine and total time of specific operation.

The complete resource statistics contains the following information: busy time, blocked time, waiting time for free space, waiting time for all incoming parts, waiting for human resource, breaks and setup time. **Busy time** is the total time that the resource spends on processing the operations. If an operation is processed and there is no free space for outgoing entities in the output buffer, this state of resource is **blocked time** of resource. One option can be used to avoid of blocked resource. In this case, the operation will not even start if there is no space for the processed entities. For some kind of operations, it is undesirable that entities stay in the resource after the operation is processed due to the unavailable space in the output buffer. If this condition is required, free space is checked before assigning the operation. If a resource cannot process the operation due to this condition, then this resource must wait (the operation is rejected) until space in buffer is released. This time interval is differentiated in the resource statistics report as **waiting time for free space**. This is a specific value of IDS.

These detailed statistics allow the implementation of *lean production* avoiding time wasting within production system. Figure 170 (left image) shows an example of some statistics. There is a time difference between the total busy time (239 minutes) and the total sum of all the operation time (235 minutes). This is due to the time of an unfinished operation that is also involved in the total busy time, in this case four time units.

Results can also be displayed as a pie graph (resources states) and a chart of processed operation (Figure 170, right image).



Simulation time lenght: 500
Warm-up time:0
Statistics:

machine_name	kind_of_proces:	busy_time	blocking_time	waiting_time_p:	waiting_time_space
Assembly_WS_2	machine	239	0	261	0
Barrelling_machine	machine	260	0	240	0
Washing_machine	machine	490	0	10	0
Assembly_WS_1	machine	355	0	145	0
*		0	0	0	0

machine_nam	name_operation	number_operatio	time_total_operation
Assembly_WS_2	assemblling_A	47	235
* Assembly_WS_2		0	0

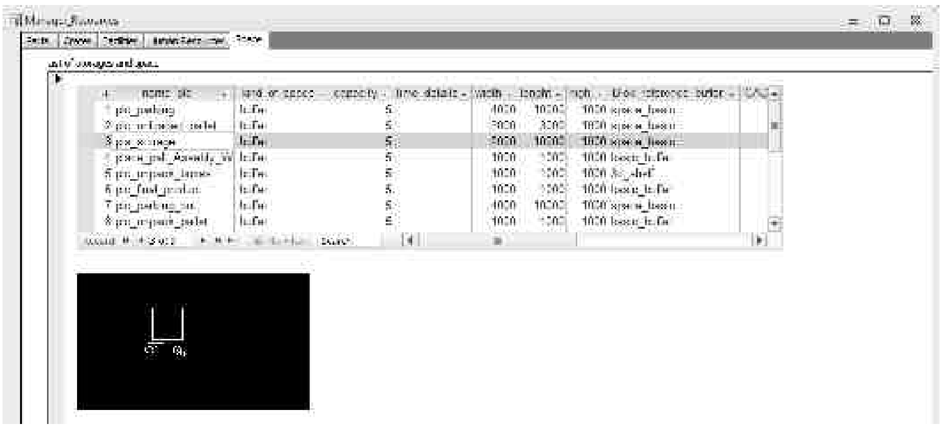
Figure 170 - Resource results

A.4.2.2 Storages and buffers

Buffers, storages or warehouses are used for storing products (entities), especially for the purposes of temporary storing of products that are waiting to be processed (buffers), storing final products in the warehouses, storing in order to cover processes uncertainties, etc.

Storages properties

Storages and buffers are defined through **IDS.DB.RESOURCE.SPACE** form. The basic properties of buffers are *unique ID number* and *name*, *kind of buffer*, *capacity*, *time details*, *dimensions* and *block reference buffer*. Property "*kind of buffer*" can have value buffer, conveyor or time path; these values define elementary space functions (e.g. *buffer* means a space used for storing entities). "*Capacity*" is the number of entities that can be at the same time in the buffer. Property "*Time details*" is used in the definition of conveyors and in cases where entities have to stay in the buffer for a defined time.



name_plc	kind_of_space	capacity	time_details	width	length	high	show_reference_buffer
1 plc_parking	buffer	5		4000	10000	1000	space_basic
2 plc_unloaded_pallet	buffer	5		5000	3000	1000	space_basic
3 plc_storage	buffer	5		5000	10000	1000	space_basic
4 place_pal_Assembly_W	buffer	5		1000	1000	1000	basic_buffer
5 plc_repaire_frames	buffer	5		1000	1000	1000	[illegible]
6 plc_final_product	buffer	5		1000	1000	1000	basic_buffer
7 plc_parking_out	buffer	5		4000	10000	1000	space_basic
8 plc_repaire_pallet	buffer	5		1000	1000	1000	basic_buffer

Figure 171 - Space input form

Additional functionalities of storages and buffers

Buffers usually have storing functions. In IDS, it can be special functions as **quantity function** in the connection with resources or **transportation function** as conveyor or time path.

Quantity function of buffers

Based on the IDS concept, one resource object represents one unit, e.g. machine. In some cases, production system involves a large number of machines the same type or those have the same properties and process the same operations. It can be a job-shop with the same production machines or mass production resources. For these cases, IDS supports a special **resource-buffer unit** with the following functions: the resource controls logic of material flows (Input rules, etc.) and buffer substitutes resource units (quantity). Entities must stay inside the buffer for a defined time, which represents the operation time (see Section 4.4.4), while the resource has zero operation time.

Setup of the IDS properties:
IDS.DB.OPERATIONSETUP.LISTOFOPERATION.*time_min_buffer* = Operation times
IDS.DB.RESOURCES.FACILITES.*kind_of_process* = "quantity_function"
IDS.DB.OPERATIONSETUP.LISTOFOPERATION.*kind_of_operation* = "production"

Practical examples are in Section 5.1 where this function is used for loading / unloading trucks with raw material. In this case, large number of trucks can process operation in the same time, independently on each other. Resource controls just the input logic of arriving trucks. Then, these trucks spend in the buffer a defined time that represents the loading and loading time of raw cement material.

Conveyor transportation system

Conveyor transportation system is also based on holding entities in the buffer for a defined time, which is equal to the movement time. This time is constant during simulation running because transporting entities have to stay on conveyor during the whole movement. In case the conveyor capacity is achieved, conveyor causes the block of resources.

Setup of the IDS properties:
IDS.DB.RESOURCES.SPACE.*kind_of_Space* = "conveyor"
IDS.DB.RESOURCES.SPACE.*time_details* = time of movement
IDS.DB.RESOURCES.SPACE.*lenght* = length of conveyor

Time path

A similar principle as in the case of conveyor is used for a function time path; entities also stay in this object for a defined time. Time path can be a route, a path, a road, an aisle, etc. This property is convenient to use together with the CAD layout containing limited material flows: by the function *"Load lengths"*, it can be loaded the length of specific flow representing the time path and then, the length of this flow can be recalculated into time (length/speed) by the function *"Times update"*. This set of functions illustrate advantages of tools integration.

The length of flows is considered from *buffer_out* point to *process_in* point. The default speed is 1 m/sec and the default lenght is 10 000 mm (10 metres). If the length or speed is zero, the default values are automatically used.

IDS.DB.TRANSPORTATION.TIMEPATH.*Load lengths()*
IDS.DB.TRANSPORTATION.TIMEPATH.*Times update()*

Setup of the IDS properties:
IDS.DB.RESOURCES.SPACE.*kind_of_space* = time_path
IDS.DB.RESOURCES.SPACE.*speed* = transportation speed
IDS.DB.RESOURCES.SPACE.*lenght* = length of path
IDS.DB.RESOURCES.SPACE.*time_details* = transporting time

This function is used in the example shown in Section 5.2 in the design of milk-runs delivering routes.

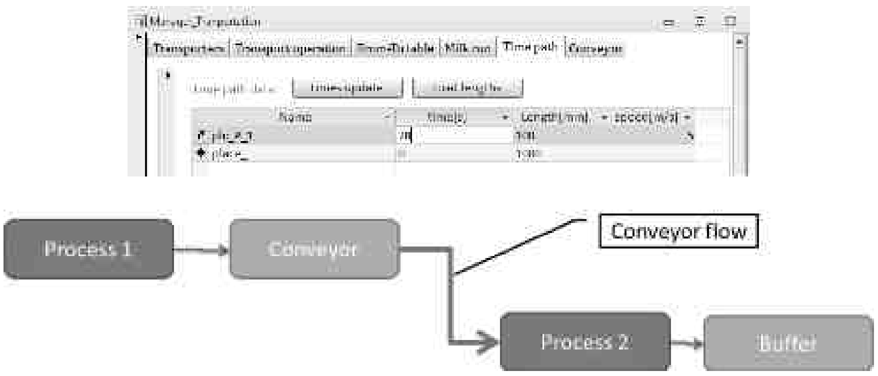


Figure 172 - Conveyor properties

Storage statistics

One of the most common tasks is to establish the space required by buffers. IDS supports it by two approaches:

- statistics description
- analysis from historical data

Statistics description of buffer consists of the average buffer size, the average waiting time, the maximum number of given parts in the buffer at any time and the time period of this maximum occupancy.

Historical data provides more detailed analysis; data are recorded by the function *"fce_Dynamic_array()"* in every buffer state changes: in the event of input or output. Recorded data set involves: buffer name, TIME, total number of parts, specific number of parts, part ID, crate ID, router and vol. Data are recorded in the external file (*"dyn_buffer_values.dat"*) after complete simulation running by a function *"function_Write_Storage_Data()"*. Data can also be recorded in DB, in tables *"Tbl_Buffer_results" and "Tbl_Buffer_dyn_data"*.

These functions and analysis are very helpful in the process of buffer definition. From those charts, it is easier to establish appropriate buffer size, i.e. space for shelves, space for pallets, etc.

Data from the exported file (*"dyn_buffer_values.dat"*) can be processed in MS Excel and the following information can be received as charts:

- chart number of all parts in the buffer during time period (occupancy)
- chart number of specific parts in the buffer during time period
- number of unique parts in the buffer within defined period (using prepared macro function (*Module_Calculation.bas*)

Setup of the IDS properties:

IDS.WITNESS.INITIALISATION.*boo Dyn write* =1 (allow data recording)
IDS.WITNESS.USERACTION.*function Write Storage Data ()*
IDS.DB.SIMULATION_RESULTS.SPACE_RESULTS.*Load dynamic occupancy value*: true/false
IDS.DB.SIMULATION_RESULTS.SPACE_RESULTS.*Load data from simulation model()*

These values are used in the buffer design described in Section 4.7.3. Example of plot of occupancy in buffers is in Table 28.

Notice:

Similar function is included in WITNESS Simulation tool, called "*Timeseries*", but with the disadvantage that data are stored just in the RAM memory without the possibility to export it.

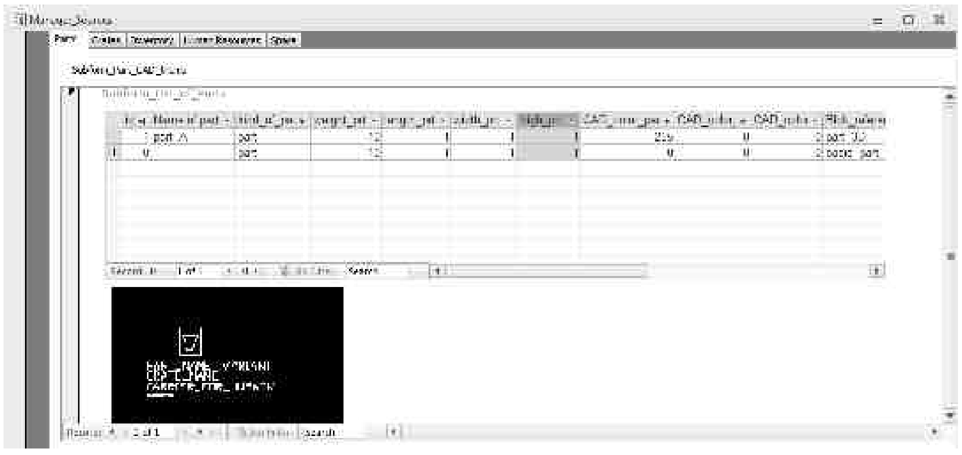
A.4.2.3 Entities and crates

Entities represent objects moving in the system and resources that can change their properties.

Crate is a property of entity that defines its actual transportation unit based on it, e.g. screws (entity) can be in a box (unpacking operation), on the pallet (transportation) or as a separate part (assembly operation).

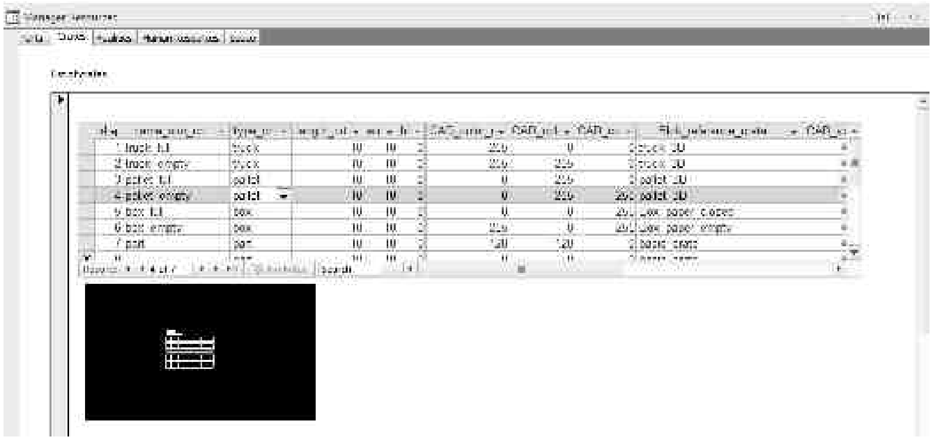
Entities properties

Entities are defined by the form **IDS.DB.RESOURCES** and parts are defined by a set of properties as: *unique ID number*, *name*, *kind of part*, *dimensions*, *weight*, *colour definition* and *block reference part*. *Kind of part* can be a standard part or dummy that is a special virtual entity that helps in the process description (see transportation helper entity in Section A.4.5). *Weight* property is helpful in the material flow design, as well as *colour definition* of them.



[illegible]	Name of part	kind_of_part	weight_pt	length_pt	width_pt	height	CAD_[illegible]	CAD_[illegible]	CAD_[illegible]	Blck_name
1	part A	part	12	1	1	1	255	0	2	part 3D
0		part	12	1	1	1	0	0	2	basic part

Figure 173 - Entities input form



id	name_group_crate	type_crate	length_of	wi	h	CAD_empty	CAD_full	CAD_[illegible]	file_reference_[illegible]
1	truck full	truck	10	10	[illegible]	225	0	0	truck 3D
2	truck empty	truck	10	10	[illegible]	225	225	0	truck 3D
3	pallet full	pallet	10	10	[illegible]	0	225	0	pallet 3D
4	pallet empty	pallet	10	10	[illegible]	0	225	[illegible]	pallet 3D
5	box full	box	10	10	[illegible]	0	0	[illegible]	box paper closed
6	box empty	box	10	10	[illegible]	225	0	[illegible]	box paper empty
7	part	part	10	10	[illegible]	120	120	0	basic crate

Figure 174 - Crates input form

Crates properties

Properties of crates are very similar to entities, as shown in the form **IDS.DB.RESOURCES.CRATE** where crates are defined. The differences are in the *type crate* used for basic definition of crates types as trucks, pallet, box or standard part (see Figure 174).

Routing of entities

Routing of entities means routes definition through system. It is based on the required operation, resources possibilities for operations, priorities, etc. Routing consists of a list of resources by which an entity can go through. This is given by data from DB and in the simulation model, it is controlled by external and internal logic functions. While external logic function (see the Section 4.4.8) selects the optimal operation for a given resource, internal logic function (see Sections 4.4.3 and 4.4.4 controls the operation in the resource and the storing in the buffer. Storing information is managed by form **IDS.DB.PLACE_SETUP.TABLE_STORAGE_MACHINE** and **IDS.DB.PLACE_SETUP.TABLE_MACHINE_STORAGE** (Figure 176). The rules are the following:

- a single resource can store entities in one specific buffer for a given operation
- a single resource can take entities from one specific buffer to a given operation
- various resources can take parts from one buffer

It is an approach for planning material flows. Material flows can be given deterministically or stochastically. In the deterministic system, routing is strictly described before the simulation is run by one possible way. In the stochastic system, entities continue to the several possible targets, based on the actual conditions of the system influenced by uncertain events such as breaks, waiting, services, random length of operation times, etc.

Setup of the IDS properties:

The assignment of operations to the resource is managed by the form **IDS.DB.WORK_POSSIBILITES**. It can be done in two ways: for a given resource assign a set of operations or for a selected operation assign a set of resources (see Figure 175, left image). It is also possible to set as a priority each pair resource-operation. Function that searches the optimal operation takes priorities into account – the operation with the higher number is selected.

One possible way to setup priorities is by using material flows lengths, i.e. distances between resources. Far resources can have lower priorities than closer resource. These lengths can be taken from the layout.

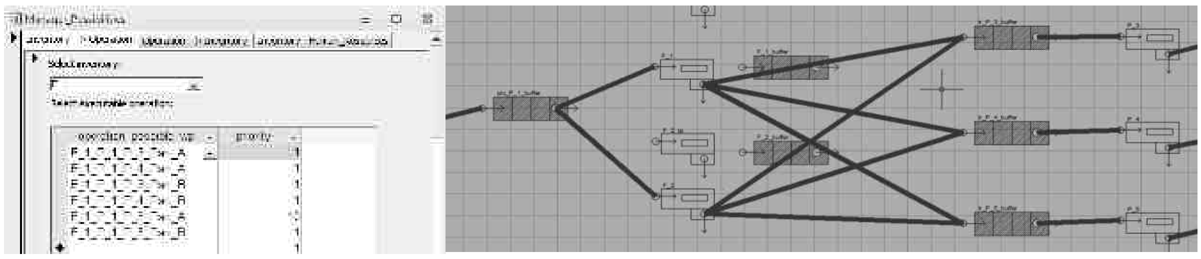


Figure 175 – Assigning operation to resources, routing map

In the AutoCAD layout drawing, it is possible to generate all possible material and transportation flows, even if they have zero value of intensity to receive complete routing map, as it is shown in Figure 175 (right image).

Notice: Searching the optimal operation can make the simulation running length longer. It is possible to switch off this property in cases that is not necessary:

IDS.WITNESS.INITIALISATION.*__boo_priority__=1* (0, searching the optimal)

The WITNESS simulation tool involves a similar function for routing. In this function, routing is given by a strict sequence of resources that an entity must go through. There is no support in assembly operations (inputting from more buffers) or possibility of selecting from more resources to process the operation.

Resource´s input and output buffer

Storing information involves the specification ***from*** that buffer entities are ***pulled*** to the resource in order to process the operation and ***to which*** buffer those entities are ***pushed*** after their process. It is controlled by forms **IDS.DB.PLACE_SETUP.TABLE_STORAGE_MACHINE** and **IDS.DB.PLACE_SETUP.TABLE_ MACHINE_STORAGE**. Input data is as simple as possible: information is automatically filled and updated by the function *"Update values"* and the user selects the buffer name from the list of buffers for a given operation. Function *"Check values"* checks all values and returns empty or wrong filled rows.

Figure 176 - Storing information

Scheduling of production

This area of industrial engineering is very extensive. In IDS, it is limited to scheduling part arrivals to the system. As mentioned in Section 4.4.2, two basic approaches are developed for the definition of entities creation: by the definition of simulation object "Part" or by the object "Part file". Figure 177 displays a setup of these schedules. Parts signed as *"Input_to_system"* in the table "Input" are automatically filled in the **IDS.DB.PART_SCHEDULING.PART_ARRIVALS** by the function *"Update_part_arrivals()"*. It is possible to define the following information:

- number of incoming parts (or "*unlimited*")
- enter time of the first part,
- time intervals between the entrance of each parts
- input batches
- entities alternatives
- sequences

Time information can be set up as stochastic functions (in object "Part") or as exact arrival time values (sequences). Values of sequences are defined in the form **IDS.DB.PART_SCHEDULING.TABLE_PART_SCHEDULING**. It contains:

- entity name
- times of entrance
- amount
- crate, router number and alternative

Then by the function **IDS.DB.PART_SCHEDULING.*Generate_part_arrivals_files()***, part files (*.par) are generated based on this data and loaded to the simulation model during its initialisation. Figure 178 shows an example of a generated schedule file. Parts are then created according to these schedules.

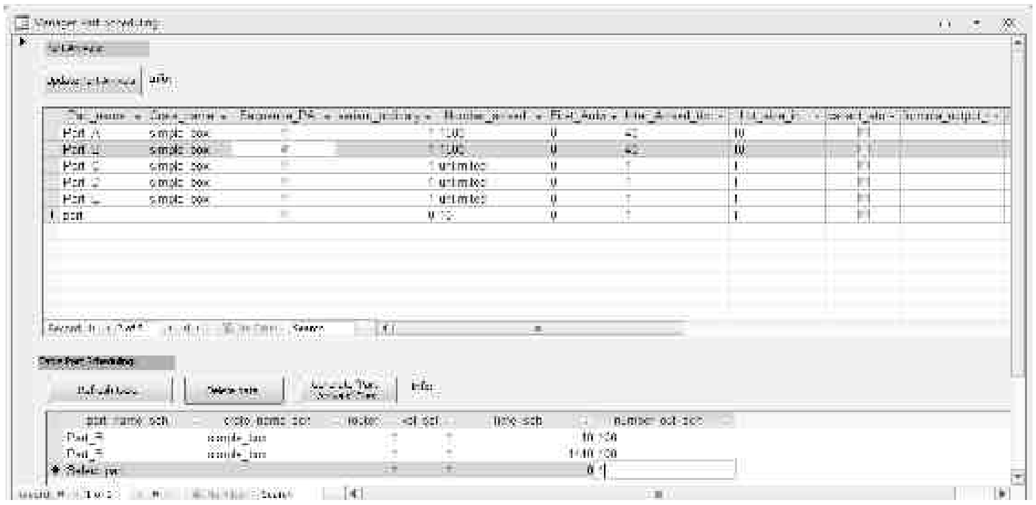
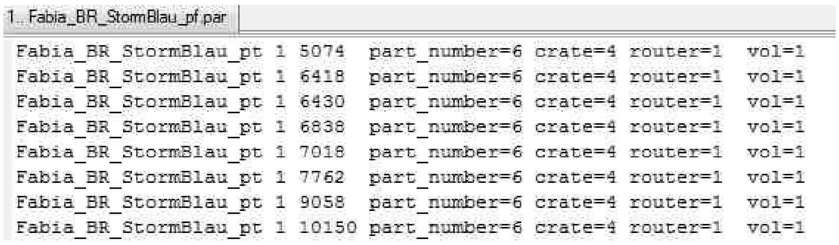


Figure 177 – Setup of part scheduling

Schedules of production are used in the practical example of paint shop in Section 5.3.



```
1_Fabia_BR_StormBlau_pf.par
Fabia_BR_StormBlau_pt 1 5074  part_number=6 crate=4 router=1  vol=1
Fabia_BR_StormBlau_pt 1 6418  part_number=6 crate=4 router=1  vol=1
Fabia_BR_StormBlau_pt 1 6430  part_number=6 crate=4 router=1  vol=1
Fabia_BR_StormBlau_pt 1 6838  part_number=6 crate=4 router=1  vol=1
Fabia_BR_StormBlau_pt 1 7018  part_number=6 crate=4 router=1  vol=1
Fabia_BR_StormBlau_pt 1 7762  part_number=6 crate=4 router=1  vol=1
Fabia_BR_StormBlau_pt 1 9058  part_number=6 crate=4 router=1  vol=1
Fabia_BR_StormBlau_pt 1 10150 part_number=6 crate=4 router=1  vol=1
```

Figure 178 - Schedule file

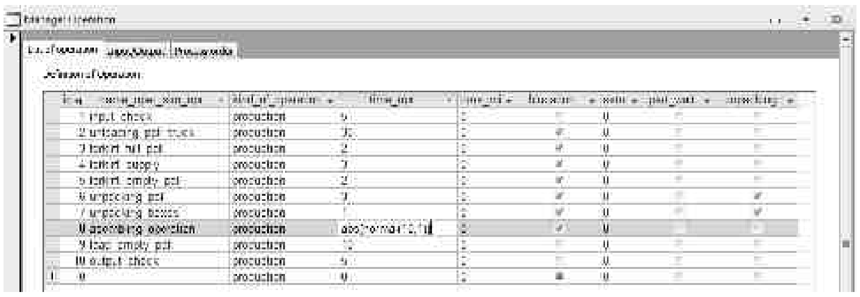
A.4.2.4 Operation

Operation properties

In the form **IDS.DB.OPERATION_SETUP**, the complete information about operation can be setup. Firstly, the definition of operation is controlled through the List_of_operations (see Figure 179) by the properties, such as *unique ID number*, *unique name*, *kind of operation*, *time_opr*, *time_min_buffer* and specific properties of operations, such as blocked state of resource (*blocation*), *setup_time*, *part_wait*, and *unpacking*. These properties are explained following sections.

This process specifies the general kind of operation that can be: *production*, *transportation*, *service*, *quantity function*, *conveyor* and *storage*. The detailed description is in the following sections: production, transportation (Section 4.7), quantity function, conveyor and storage (Section A.4.2).

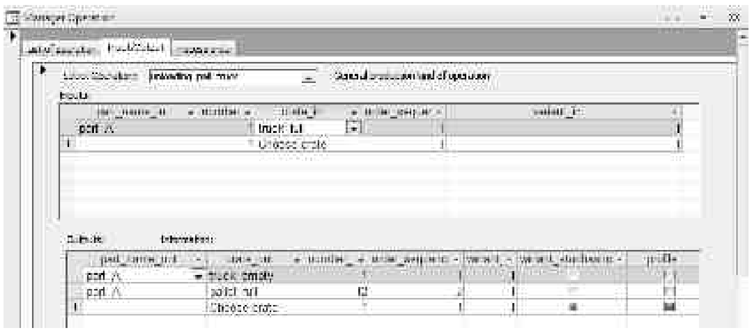
Time_opr defines an operation time as well as *time_min_buffer* for cases of storage of kind of operations.



[illegible]	[illegible]	kind_of_operation	time_[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	unpacking
1	input check	production	5	[illegible]		0		
2	unloading pal truck	production	30	[illegible]	✓	0		
3	forklift full pal	production	2	[illegible]	✓	0		
4	forklift supply	production	3	[illegible]	✓	0		
5	forklift empty pal	production	2	[illegible]	✓	0		
6	unpacking pal	production	3	[illegible]	✓	0		✓
7	unpacking boxes	production	1	[illegible]	✓	0		✓
8	assembling operation	production	abs(normal(2,1))	[illegible]	✓	0		
9	load empty pal	production	10	[illegible]		0		
10	output check	production	5	[illegible]		0		
0		production	0	[illegible]	■	0		

Figure 179 - Operation definition form

Each operation consists of input and output sequences of entities as it can be defined in the form **IDS.DB.OPERATION_SETUP.INPUT_OUTPUT** (see Figure 180). Each sequence consists of values, such as part name, crate name, order in sequence and variant, output sequence, which can have output stochastic function definition.



Select Operation: unloading pal truck

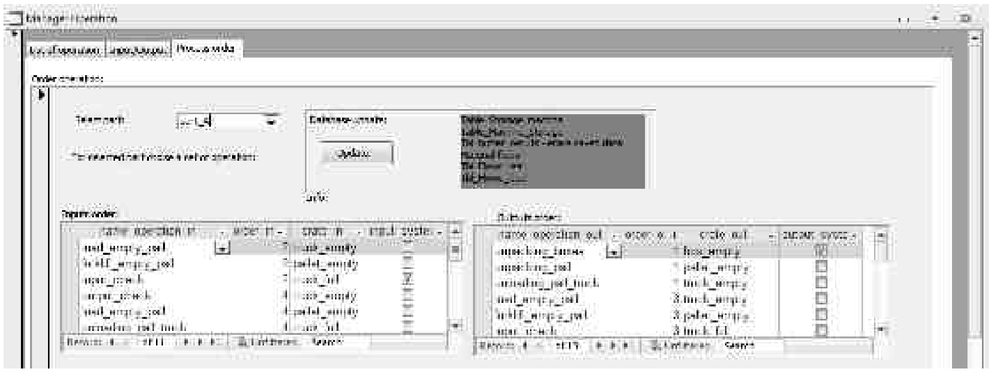
Set kind production kind of operation

part_name_in	number	crate_in	order_sequence	variant_in
part A	1	truck full	1	1
	1	Choose crate	1	1

part_name_out	crate_out	number	order_sequence	variant	variant_stochastic	profile
part A	truck empty	1	1	1		
part A	pallet full	12	2	1		
	Choose crate	1	1	1	■	■

Figure 180 - Operation input/output definition form

By the form **IDS.DB.OPERATION_SETUP.PROCESS_ORDER** (see Figure 181), sequences of operation in the system can be established, the so called ***technologic process***. The user fills the order of the operation sequence for the selected entity and also establishes *"Input to system"* and *"Output to system"* property for entities incoming/outgoing in the system.



Select part: part_A

name_operation_in	order_in	crate_in	input_system
rail_empty_pal	1	crate_empty	
forklift_empty_pal	2	pallet_empty	
input_check	3	truck_full	☒
output_check	4	crate_empty	
rail_empty_pal	4	pallet_empty	
unloading_pal_truck	4	crate_full	

name_operation_out	order_out	crate_out	output_system
unpacking_boxes	1	box_empty	☒
unpacking_pal	1	pallet_empty	☐
unloading_pal_truck	1	truck_empty	☐
rail_empty_pal	3	truck_empty	☐
forklift_empty_pal	3	pallet_empty	☐
input_check	3	truck full	☐

Figure 181 - Sequence of operation

Unpacking operation

This function is prepared for unpacking a kind of operation. For example: a pallet is loaded by several boxes, after the unpacking operation, this pallet can only be taken away when all boxes are removed and the pallet is empty. A schema of this process is Figure 182. Simulation model involved functions (*"function_Unpack_previous"*, *"function_Unpack_next"*) that check a condition:

```
((number pallets x capacity of pallet)-(number of boxes)) / capacity of
pallet ≥ number transporting pallets
```

If it is true, then this pallet (carrier) can continue its process, e.g. be taken by a forklift.

Setup of the IDS properties:
IDS.DB.OPERATION_SETUP.LIST_OF_OPERATION.*unpacking* = true
IDS.DB.OPERATION_SETUP.*Output.carrier* = true (one of the outgoing entity must be *"carrier"*)

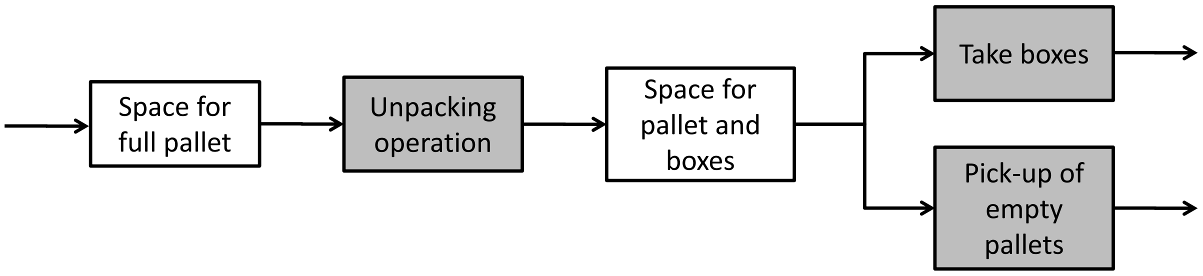


Figure 182 - Unpacking operation schema

Part-wait function

This function allows waiting for the rest of the inputting entities. The IDS functions check numbers of inputting entities that are waiting in the buffers before the operation is selected by the external logic control (see Section 4.4.8.). With this function, it is allowed to select and assign an operation to the resource, even though there is no appropriate number of entities. It can be helpful in some specific operations, such as loading parts, etc.

Setup of the IDS properties:
IDS.DB.OPERATION_SETUP.LIST_OF_OPERATION.*part_wait* = true

Waiting for free space in the output buffers

This function checks the available space in the output buffers. If there is not enough space, the operation is denied. This can be used, for example, in the transportation of heavy loads by crane. As it is impossible to hold a load for a long period of time for technological and security reasons, crane drivers must check the free space of the destination before the operation. Other example can be a factory gate that should not be blocked due to the constant movement. In the statistics of resource, this state is described as *"waiting_time_space"*.

Setup of the IDS properties:
IDS.DB.OPERATION_SETUP.LIST_OF_OPERATION.*blocation*=true

A.4.3 Simulation functionalities

A.4.3.1 Generation of simulation models

The automatic generation of simulation models is described in Section 4.4.1. The process can be controlled by functions in the form **IDS.DB.GENERATION_SIM_MODELS** (Figure 183).

Before the generating process, all data are checked (function *"Check_DB_values()"*) and prepared (*"Preparing data()"*). Only after that, the model can be generated by function *"Generate()"*.

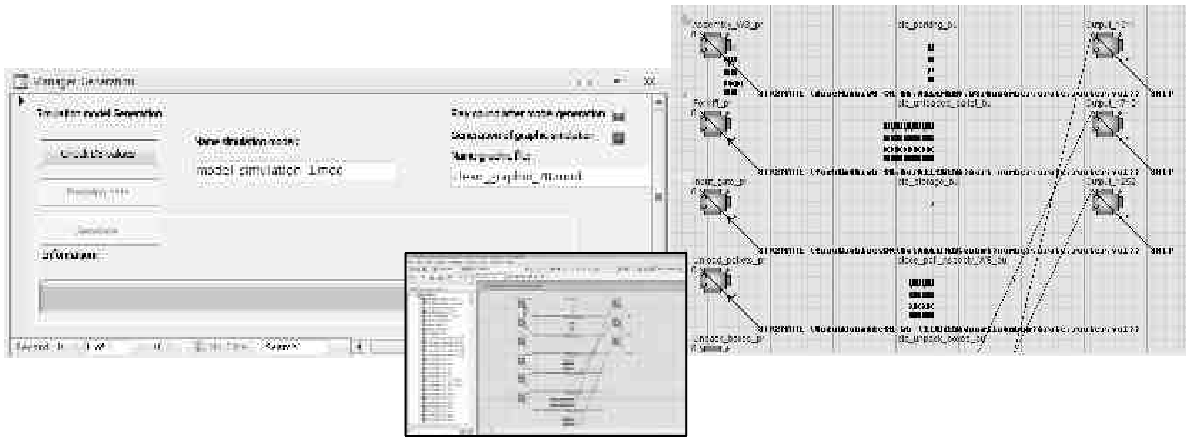


Figure 183 - Form: Generation of simulation models and model

A.4.3.2 Simulation model initialisation and warm-up

After the automatic initialisation, the generated model is ready to be run (Section 4.4.7). In some cases, it is important to cut statistics information off from the beginning of the simulation (Time = 0) until the time when a simulation model contains a part-in-process and when the system is stable to receive model's statistics and behaviour corresponding to the reality. This period is called **warm-up** (see example in Section 5.3.5).

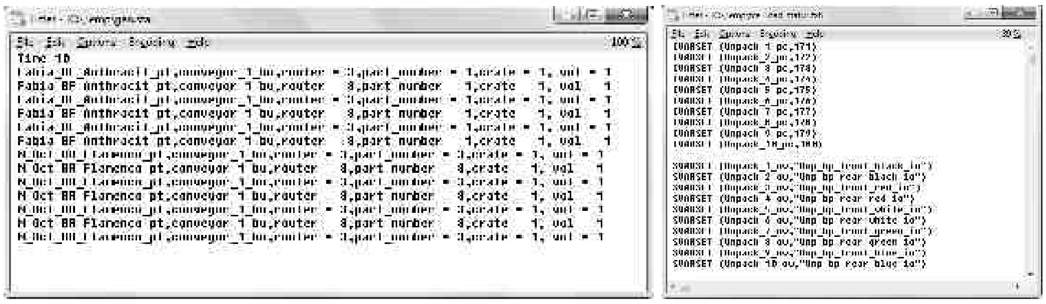
In the IDS tool, two possibilities are available to achieve it. The first possibility is to run the simulation model into warm-up time, reset statistics and then continue in simulation. The second option is to load the initialisation file containing the information about resource states, WIP, etc. that can be achieved in the warm-up time.

Function **IDS.WITNESS.USER_ACTION**: ***fce Reset Statistics()*** resets all data of material flows, throughput, etc.; and statistics of resources, buffers and human resources are cleared by a function "*ResetReportsByType ()*".

Other option is based on loading the external file. This file is generated by macros and functions involved in MS Excel. These functions prepare sheets with data about simulation objects. The user can specify their states, such as resources states (busy, idle, break), statistics of processed operations, number of parts in the buffers, etc. After that, files (**.sta*) are generated and loaded into the simulation model.

This function can be helpful also for definition unusual states in the system.

Entities in the resources and buffers are loaded by function *"ImportState()"* and data about entities in the buffers are loaded by a function *"fce_Load_status()"*. Examples of these files are in Figure 184, and the codes are in Appendix (Section A.3.3.3).



```
Time 10
Fabia_BF_Anthracit_pt,conveyor_1_bu,router = 3,part_number = 1,crate = 1, vol = 1
Fabia_BF_Anthracit_pt,conveyor_1_bu,router = 3,part_number = 1,crate = 1, vol = 1
Fabia_BF_Anthracit_pt,conveyor_1_bu,router = 3,part_number = 1,crate = 1, vol = 1
Fabia_BF_Anthracit_pt,conveyor_1_bu,router = 3,part_number = 1,crate = 1, vol = 1
Fabia_BF_Anthracit_pt,conveyor_1_bu,router = 3,part_number = 1,crate = 1, vol = 1
Fabia_BF_Anthracit_pt,conveyor_1_bu,router = 3,part_number = 1,crate = 1, vol = 1
N_Oct_BR_Florence_pt,conveyor_1_bu,router = 3,part_number = 3,crate = 1, vol = 1
N_Oct_BR_Florence_pt,conveyor_1_bu,router = 3,part_number = 3,crate = 1, vol = 1
N_Oct_BR_Florence_pt,conveyor_1_bu,router = 3,part_number = 3,crate = 1, vol = 1
N_Oct_BR_Florence_pt,conveyor_1_bu,router = 3,part_number = 3,crate = 1, vol = 1
N_Oct_BR_Florence_pt,conveyor_1_bu,router = 3,part_number = 3,crate = 1, vol = 1
N_Oct_BR_Florence_pt,conveyor_1_bu,router = 3,part_number = 3,crate = 1, vol = 1
N_Oct_BR_Florence_pt,conveyor_1_bu,router = 3,part_number = 3,crate = 1, vol = 1
```

```
IVARSET (Unpack_1_pc,171)
IVARSET (Unpack_2_pc,172)
IVARSET (Unpack_3_pc,173)
IVARSET (Unpack_4_pc,174)
IVARSET (Unpack_5_pc,175)
IVARSET (Unpack_6_pc,176)
IVARSET (Unpack_7_pc,177)
IVARSET (Unpack_8_pc,178)
IVARSET (Unpack_9_pc,179)
IVARSET (Unpack_10_pc,180)

SVARSET (Unpack_1_ou,"Unp_bp_front_black_io")
SVARSET (Unpack_2_ou,"Unp_bp_rear_black_io")
SVARSET (Unpack_3_ou,"Unp_bp_front_red_io")
SVARSET (Unpack_4_ou,"Unp_bp_rear_red_io")
SVARSET (Unpack_5_ou,"Unp_bp_front_white_io")
SVARSET (Unpack_6_ou,"Unp_bp_rear_white_io")
SVARSET (Unpack_7_ou,"Unp_bp_front_green_io")
SVARSET (Unpack_8_ou,"Unp_bp_rear_green_io")
SVARSET (Unpack_9_ou,"Unp_bp_front_blue_io")
SVARSET (Unpack_10_ou,"Unp_bp_rear_blue_io")
```

Figure 184 - Initialising of model from the files

A.4.3.3 WITNESS coordinates function

As mentioned above, the generation of model was developed for background simulation running. Objects are generated in WITNESS simulation model in alphabetic order. For a better identification of objects and make model more clear, there is the function **IDS.WITNESS.USER_ACTION**. ***fce Position objects (icon size, zoom).*** This function loads the coordinates of objects in AutoCAD and places the objects in the simulation model based on them, as shown in Figure 185. There are the layout from CAD (first image), generated objects in alphabetic order (second image) and objects positioned according to the layout (third image).

It is also possible to import complete simplified drawing of a layout in the graphic format dxf, exported from AutoCAD (**IDS.CAD.FACILITY. WITNESS.*Save dxf file().***

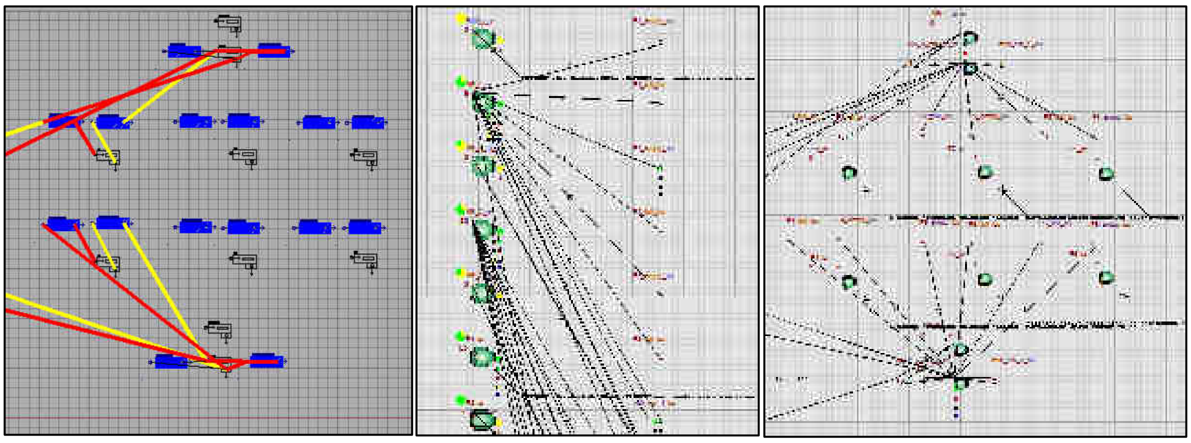


Figure 185 – WITNESS coordinates function

A.4.3.4 Stochastic profiles

Stochastic function can be defined by a distribution of functions (e.g. *Normal (45,7)*) or by profiles. For this purpose, there is the form **IDS.DB.STOCHASTIC. PROFILES** based on the user's data can be defined, for a given integer value is assigned its weight as shown in Figure 186. It can be helpful for the definition of output quality, random material flows, etc.

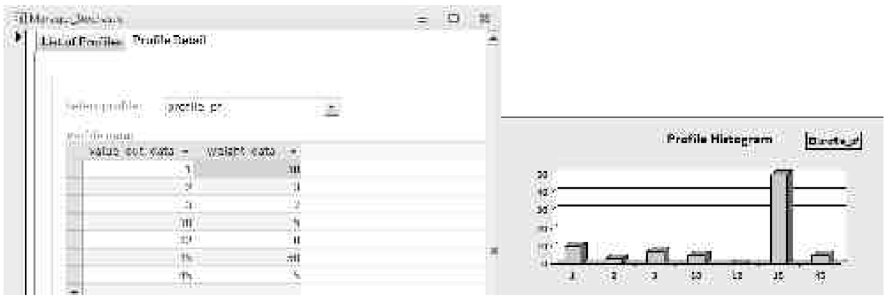


Figure 186 - Stochastic profiles

A.4.4 Material flows

Material flows (MF) are a very important part in the factory and in the facility layout design. MF can be automatically generated which helps the user with the design, as explained in Section 4.5.3. IDS involves user forms that assist the control of work with material flows. It can be **IDS.CAD.FACILITY.MATERIAL_FLOWS** for a work in facility design or IDS.CAD.FACTORY, which involves functions for generating flows in the factory – see next Section A.4.4.

IDS.CAD.FACILITY.MATERIAL_FLOWS has two sections: Generating and Displaying of MF (see Figure 187).

A.4.4.1 Generation of material flows

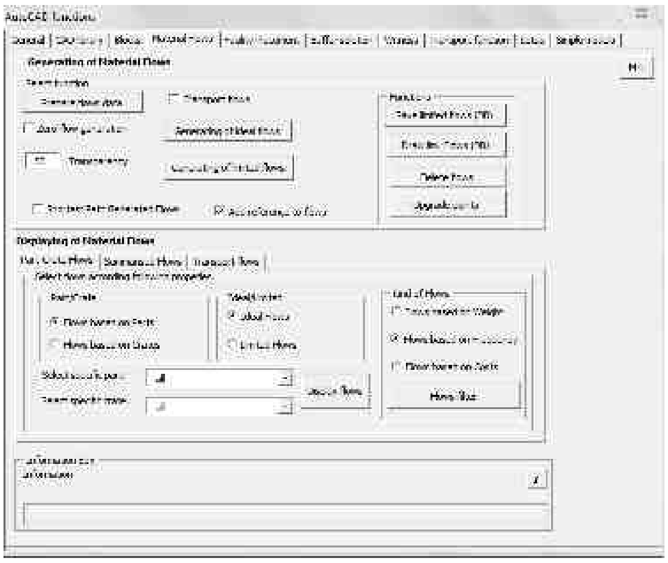
Functions involved in this section generate material flows (see Figure 187). Firstly, data must be arranged by the function *"Prepare_flows_data()"*. After that, **ideal** or **limited flows** can be generated by appropriate functions with several parameters:

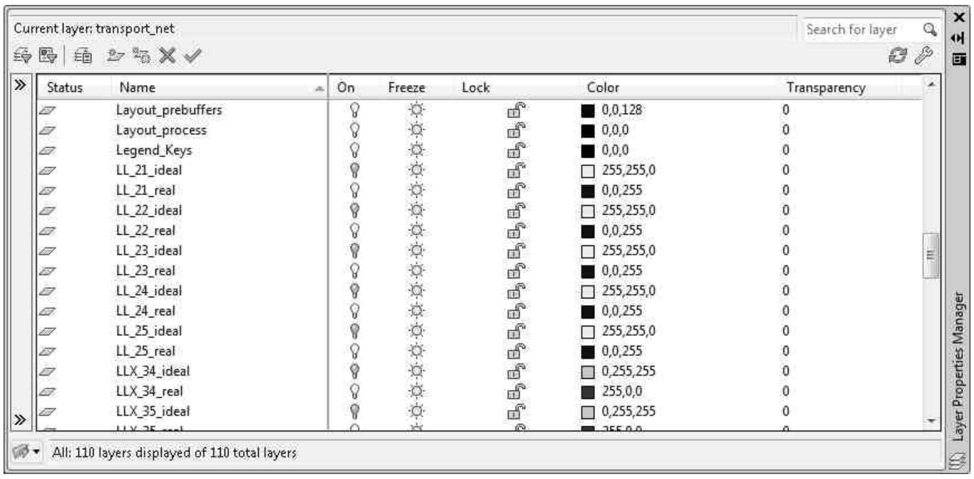
- **IDS.CAD.FACILITY.MATERIAL_FLOWS.**<u>*Zero-flow generation*</u>
- **IDS.CAD.FACILITY.MATERIAL_FLOWS.**<u>*Transparency*</u>
- **IDS.CAD.FACILITY.MATERIAL_FLOWS.**<u>*Add reference to flows*</u>
- **IDS.CAD.FACILITY.MATERIAL_FLOWS.**<u>*Transport flows*</u>
- **IDS.CAD.FACILITY.MATERIAL_FLOWS.**<u>*Shortest path generated flows*</u>

By the option *"Zero-flow generation"* = true, flows are generated with the default value 50, even if they have zero of width, e.g. before running a simulation experiment that provides values of material flows. MF can be generated as a percentage of **transparency** and also as reference containing flow information (product name, crate etc.). **Transport flows** lead directly from the start station to the final destination of transportation omitting the transporter buffer. The **shortest path flows** are generated based on the transportation network and based on finding the shortest path between given start and final station.

Other functions (**IDS.CAD.FACILITY.MATERIAL_FLOWS**) are:

- *"Saving_limited_flows()"* – save complete data of polyline
- *"Delete_flows()"* – delete all flows in the layout
- *"Draw_flows()"* – draw flows based on data from DB
- *"Upgrade_points()"* – upgrading coordinates of objects




Current layer: transport_net

Search for layer

Name	Color	Transparency
Layout_prebuffers	■ 0,0,128	0
Layout_process	■ 0,0,0	0
Legend_Keys	■ 0,0,0	0
LL_21_ideal	□ 255,255,0	0
LL_21_real	■ 0,0,255	0
LL_22_ideal	□ 255,255,0	0
LL_22_real	■ 0,0,255	0
LL_23_ideal	□ 255,255,0	0
LL_23_real	■ 0,0,255	0
LL_24_ideal	□ 255,255,0	0
LL_24_real	■ 0,0,255	0
LL_25_ideal	□ 255,255,0	0
LL_25_real	■ 0,0,255	0
LLX_34_ideal	▨ 0,255,255	0
LLX_34_real	■ 255,0,0	0
LLX_35_ideal	▨ 0,255,255	0

All: 110 layers displayed of 110 total layers

Layer Properties Manager

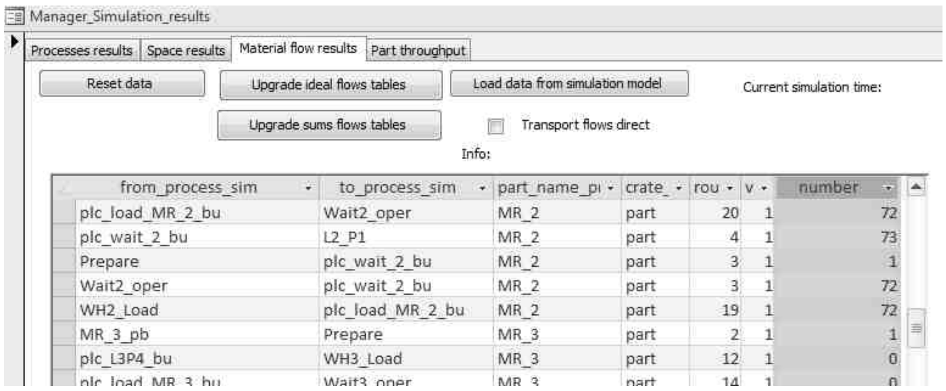
Figure 187 – Material flows processing

A.4.4.2 Displaying of material flows

As referred above, each material flow belongs to a specific layer based on an entity name, crate kind, sums of flows, etc. It is possible to switch on/off those layers manually or by filtering function provided by IDS, according to the user's needs. Based on the classification of flows (Table 12), the following types of flows are determinable: *"Part and Crate"*, *"Summarised flows"*, *"Transport flows"*, *"Ideal"* or *"Constrained"*. These flows can be more specific – they display flows for a selected part or for all parts. Flows can be based on weight, frequency or costs. Transportation flows can have colour based on the intensity of pallet colours or one colour shade.

A.4.4.3 Loading values from simulation

The number of entities going through resources and buffers are basic results received from simulation experiment. Based on this, the width of the material flows are calculated and generated in the layout. In order to load and manage it, there is the form **IDS.DB.SIMULATION_RESULTS.MATERIAL_FLOW_RESULTS** in DB that contains a set of functions (see Figure 188). The function *"Upgrade_ideal_flows_tables()"* prepares a list of material flows and updates them in DB. After that, data from simulation model can be loaded by the function *"Load data_from_simulation_model()"* and sums between objects can be calculated by *"Upgrade sums flows tables()"*. The function *"Reset_data()"* cleans this table.



from_process_sim	to_process_sim	part_name_pı	crate_	rou	v	number
plc_load_MR_2_bu	Wait2_oper	MR_2	part	20	1	72
plc_wait_2_bu	L2_P1	MR_2	part	4	1	73
Prepare	plc_wait_2_bu	MR_2	part	3	1	1
Wait2_oper	plc_wait_2_bu	MR_2	part	3	1	72
WH2_Load	plc_load_MR_2_bu	MR_2	part	19	1	72
MR_3_pb	Prepare	MR_3	part	2	1	1
plc_L3P4_bu	WH3_Load	MR_3	part	12	1	0
plc_load_MR_3_bu	Wait3_oper	MR_3	part	14	1	0

Figure 188 - Material flows managing

A.4.4.4 Material flow analysis

IDS contains a set of tools for material flow analysis (see Figure 205). Using functions in the form **IDS.DB.MATERIAL_FLOW_ANALYSE**, total costs of all material flows in the layout are calculated, as the multiplication of flow length, intensity and costs. These sums can be calculated for a chosen part (entity) or a kind of flow (ideal or constrained). This summary can be displayed as a P-Q analysis or as a pie graph. All information flows is displayed in the table as well as the sums for each flow. An example of use is in Section A.4.8.

A.4.5 Space limitation

IDS allows inserting the appropriate number of entities block into the layout. This number is given by **IDS.DB.SIMULATION_RESULTS.SPACE_RESULTS.*<u>Actual value</u>***. By the form in CAD **IDS.CAD.FACILITY.BUFFER_SOLUTION**, it is possible to insert blocks automatically (function *"Insert part-*

in-block into layout()"). The user can arrange blocks in a given space of buffer by moving, rotating, etc. If there is no space for some of them, they must be deleted. This new buffer limitation can be saved in DB (function *"Save_limits()"*).

Other functions that can be useful are: *"Save_current_position()"* for saving coordinates of blocks, *"Delete all()"* for deleting (clearing) all part-blocks and *"Load_position_from_DB()"* by those blocks that can be inserted again.

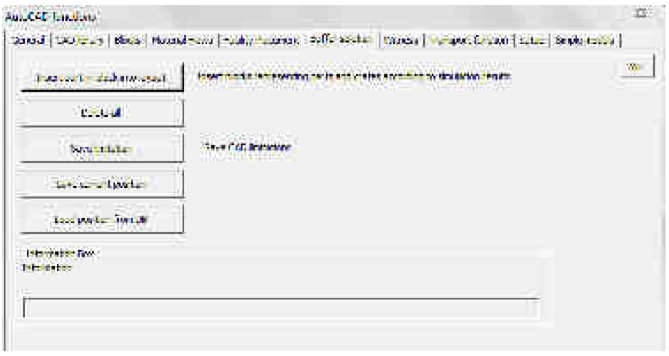
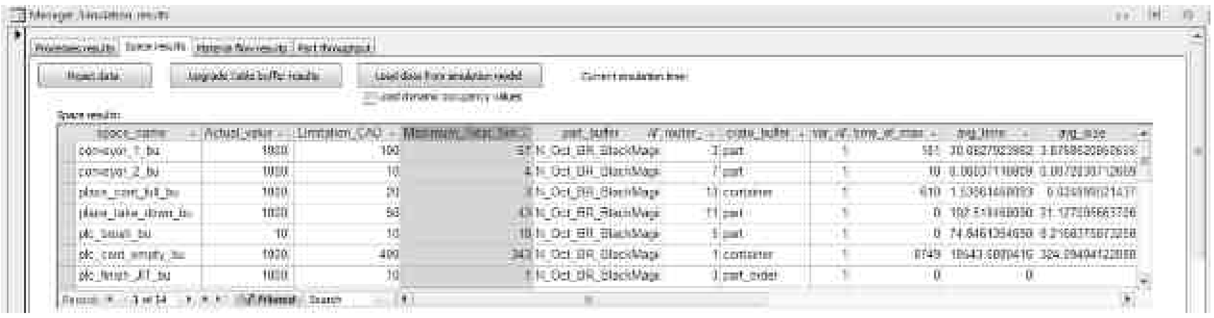

AutoCAD functions
General | CAD library | Blocks | Material flow | Quality | Buffer | Stations | Transport | Setup | Simple routing
Insert part-blocks into layout
Delete all
Save limitation
Save current position
Load position from DB
Insert blocks representing parts and pallets according to simulation results
Save CAD limitations
Information flow
Information

Figure 189 - Form: Inserting entities blocks

The results from the simulation experiments and the CAD layout are available in the form **IDS.DB.SIMULATION_RESULTS.SPACE_RESULTS** (see Figure 190). This form contains the complete statistics for a given buffer (average buffer size, average waiting time, the maximum number of given parts and time period of this maximum occupancy).


Manager_Simulation results
Processes results | Space results | Material flow results | Part throughput
Import data
Upgrade table buffer results
Load data from simulation model
Current simulation time:
Load dynamic occupancy values
Space results:

space_name	Actual_value	Limitation_CAD	Maximum_Total_Sim	part_buffer	Nº_router	code_buffer	var_nº	time_of_max	avg_time	avg_size
conveyor_1_bu	1000	100	37	N_Oct_BR_BlackMagi	3	part	1	181	30.0827923862	3.0768620862633
conveyor_2_bu	1000	10	4	N_Oct_BR_BlackMagi	7	part	1	10	0.00037118929	0.0072030712609
place_cont_full_bu	1000	20	[illegible]	N_Oct_BR_BlackMagi	11	container	1	610	1.13561468093	0.034899121477
place_take_down_bu	1000	50	43	N_Oct_BR_BlackMagi	11	part	1	0	192.51988030	31.177205663736
plc_small_bu	10	10	10	N_Oct_BR_BlackMagi	5	part	1	0	74.6461364050	8.2168375873258
plc_cont_empty_bu	1000	400	343	N_Oct_BR_BlackMagi	1	container	1	8749	18643.5870416	324.09494122888
plc_finish_JIT_bu	1000	10	1	N_Oct_BR_BlackMagi	3	part_order	1	0	0	

Record: 1 of 14 No Filter Search

Figure 190 - Buffer results form

IDS provides an approach for the **buffer design** and establishes the **optimal size**, as mentioned in Section 4.7.3 and in Figure 108.

For this purpose, the properties *"Actual value"*, *"Limitation_CAD"* and *"Maximum_Total_Sim"* are used in the form **IDS.DB.SIMULATION_RESULTS.SPACE_RESULTS**. The buffer size is controlled by them in the different software tools. *Limitation_CAD* values are received from the CAD layout. It is the maximum number of a given part that is possible to arrange in the given buffer space. *Maximum_Total_Sim* are limitations from simulation experiments. *Actual value* is a buffer size used as an actual system value; it can be the current value of limitation in the simulation or in the CAD layout.

A.4.6 Factory layouts

IDS provides a set of functions for a factory design. They are involved in the forms **IDS.CAD.FACTORY** and **IDS.DB.FACTORY_LAYOUT_DESIGN**. IDS supports three approaches in factory design: *CRAFT* and *Graph Theory Block Layout,* based on material flows, and *CORELAP,* based on relationships. By those algorithms, layouts can be automatically generated.

Input factory data

All required data are involved in the form **IDS.DB.FACTORY_LAYOUT_DESIGN**. There is a list of departments defined by a unique name, an ID number and colour. Layout alternatives are designed based on the arrangement of these departments. Alternatives are established by a *unique name*, *width* of factory, *length* of factory, *number of departments*, *maximal width of department*, *AEIOUX* and *alpha coefficients* (CORELAP). Each alternative is configured by the form **IDS.DB. FACTORY_LAYOUT_DESIGN.ALTERNATIVE_DESIGN** where is possible to setup the department's sizes and the sequence number. The size of the department is recalculated to the appropriate number of units, while the sequence number is an order of placing a given department during the design, based on importance, size, material flows, etc. or calculated by algorithms (see described heuristics in Section 2.2.1. Inputting sequence can be setup in the form **IDS.DB.FACTORY_LAYOUT_DESIGN.SEQUENCE**.

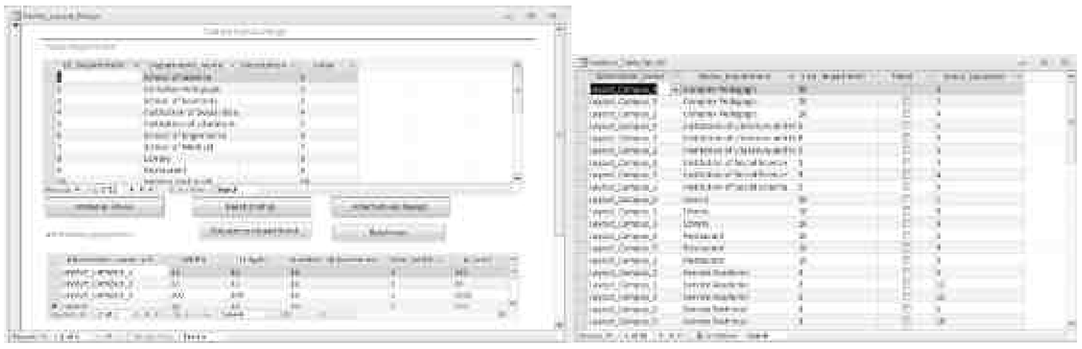
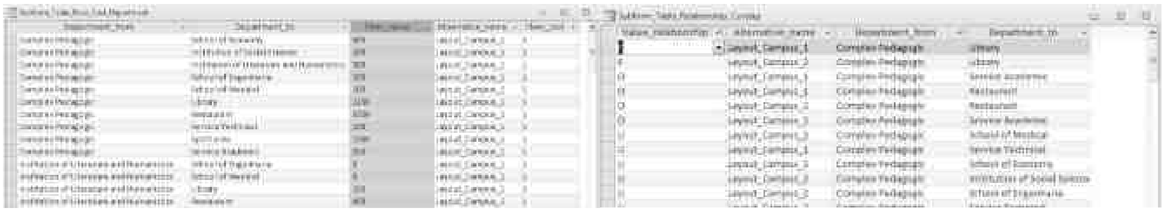


Figure 191 – Form: Factory layout design and alternative configuration

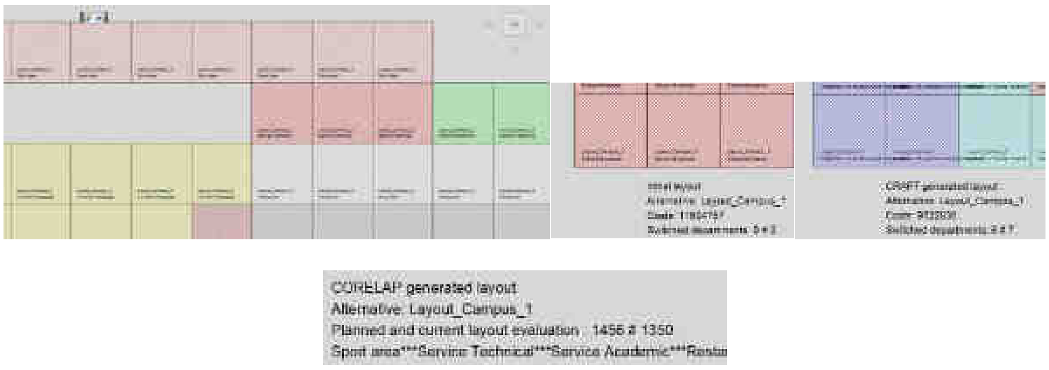
As mentioned, basic input data sets are material flows (**IDS.DB. FACTORY_LAYOUT_DESIGN.MATERIAL_FLOWS**) and relationships (**IDS.DB.FACTORY_LAYOUT_DESIGN. RELATIONSHIPS**) that are displayed in Figure 192. These two forms corresponds to the *"from-to"* table and *"relationship chart"*. Both of them can also be configured by different values for each alternative.



Department_from	Department_to	[illegible]	Alternative_name	[illegible]
Complex Pedagogic	School of Economy	[illegible]	Layout_Campus_1	[illegible]
Complex Pedagogic	Institution of Social Science	[illegible]	Layout_Campus_1	[illegible]
Complex Pedagogic	Institution of Literature and Humanities	[illegible]	Layout_Campus_1	[illegible]
Complex Pedagogic	School of Engineering	[illegible]	Layout_Campus_1	[illegible]
Complex Pedagogic	School of Medical	[illegible]	Layout_Campus_1	[illegible]
Complex Pedagogic	Library	[illegible]	Layout_Campus_1	[illegible]
Complex Pedagogic	Restaurant	[illegible]	Layout_Campus_1	[illegible]
Complex Pedagogic	Service Technical	[illegible]	Layout_Campus_1	[illegible]
Complex Pedagogic	Sport area	[illegible]	Layout_Campus_1	[illegible]
Complex Pedagogic	Service Academic	[illegible]	Layout_Campus_1	[illegible]
Institution of Literature and Humanities	School of Engineering	[illegible]	Layout_Campus_1	[illegible]
Institution of Literature and Humanities	School of Medical	[illegible]	Layout_Campus_1	[illegible]
Institution of Literature and Humanities	Library	[illegible]	Layout_Campus_1	[illegible]
Institution of Literature and Humanities	Restaurant	[illegible]	Layout_Campus_1	[illegible]

Values_relationship	Alternative_name	Department_from	Department_to
[illegible]	Layout_Campus_1	Complex Pedagogic	Library
E	Layout_Campus_2	Complex Pedagogic	Library
O	Layout_Campus_1	Complex Pedagogic	Service academic
O	Layout_Campus_1	Complex Pedagogic	Restaurant
O	Layout_Campus_2	Complex Pedagogic	Restaurant
O	Layout_Campus_1	Complex Pedagogic	Service Academic
U	Layout_Campus_1	Complex Pedagogic	School of Medical
U	Layout_Campus_1	Complex Pedagogic	Service Technical
U	Layout_Campus_1	Complex Pedagogic	School of Economy
U	Layout_Campus_2	Complex Pedagogic	Institution of Social Science
U	Layout_Campus_2	Complex Pedagogic	School of Engineering

Figure 192 – Forms: Material flows and Relationships



Initial layout
Alternative: Layout_Campus_1
Costs: 1160475?
Switched departments: 3 # 2

CRAFT generated layout
Alternative: Layout_Campus_1
Costs: 972830
Switched departments: 2 # 7

CORELAP generated layout
Alternative: Layout_Campus_1
Planned and current layout evaluation : 1456 # 1350
Sport area***Service Technical***Service Academic***Resta

Figure 193 – Generated units

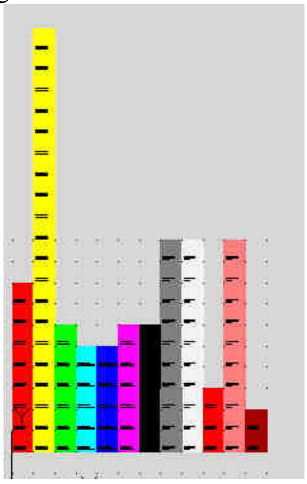


Figure 194 – Manual

A.4.7 Factory design in CAD

Factory layout alternative design is controlled by the form **IDS.CAD.Factory**. The information about the selected alternative is loaded by **IDS.CAD.Factory.Alternative_ settings:** ***Load/Reload data from DB()*** (see Figure 195). Through this form, configuration can be changed or a new alternative added as well (function *"Add_new()"*).

Four different ways can be used to design a layout alternative: manually, CRAFT, CORELAP and Graph Theory Block Layout. Those layouts consist of department unit blocks. Each generated unit (square block) contains a text string with department and layout alternative information. Units are distinguished by colours based on departments. For each generated layout, an informative legend is inserted, involving the achieved score of the current alternative and the maximum possible (theoretical) score for comparison. There is also information involved as alternative name, switched departments, used algorithm, etc. (see examples in Figure 193).

Automatically generated layouts usually need improvements as the adjustment of some block units to a better use of space or joining the split units together.

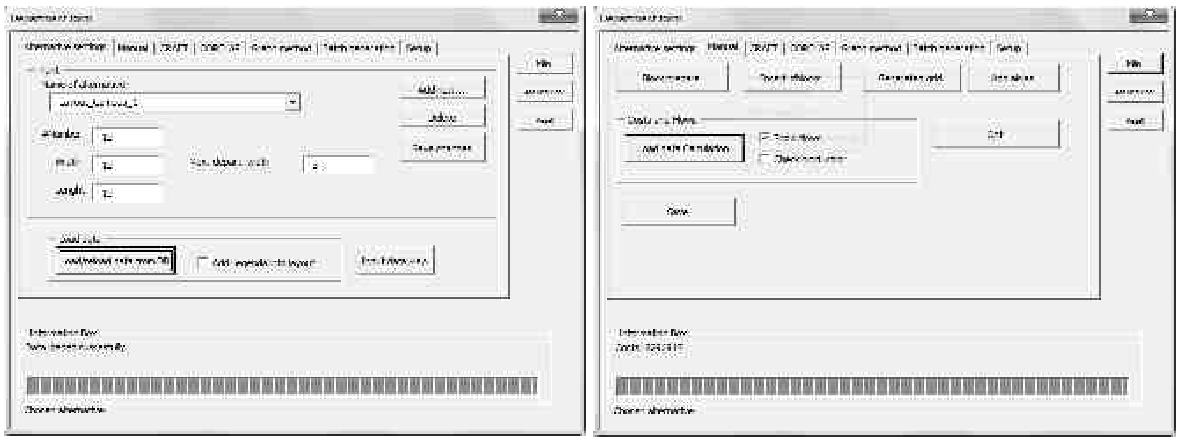


Figure 195 – Form: Alternative configuration and Manual design form

A.4.7.1 Manually designed layout

Form **IDS.CAD.FACTORY.MANUAL** provides several functions helping manual design alternatives (see Figure 195). It means that the arrangement of department units is done by the user. Firstly, a unit block must be loaded into a layout drawing (function *"Block_prepare()"*) and *grid points* (function *"Generate grid()"*) as the preparation to insert the appropriate number of department unit blocks (function *"Insert #blocks()"*). Then, the user can arrange those unit blocks into the factory layout where generated snap points help to arranging them in the grid. The layout effectiveness can be calculated by the total cost of material flows (function *"Load_data_Calculation()"*). Material flows are automatically generated within this function and upgraded during each change. By this function, different alternatives can be evaluated and the optimal solution can be easily selected. This manually designed layout can be used as an initial solution for CRAFT optimization (function *"Opt()"*).

A.4.7.2 CRAFT

CRAFT based alternatives can be generated by a function in the form **IDS.CAD.FACTORY.CRAFT**. The implementation is described in Section 4.5.3. Material flows can be calculated as ***Cartesian*** or ***Euclidian***. A random initial solution can also be generated (function *"Random_layout_generate()"*) and CRAFT tries to improve it (function *"Optimize_current_layout()"*).

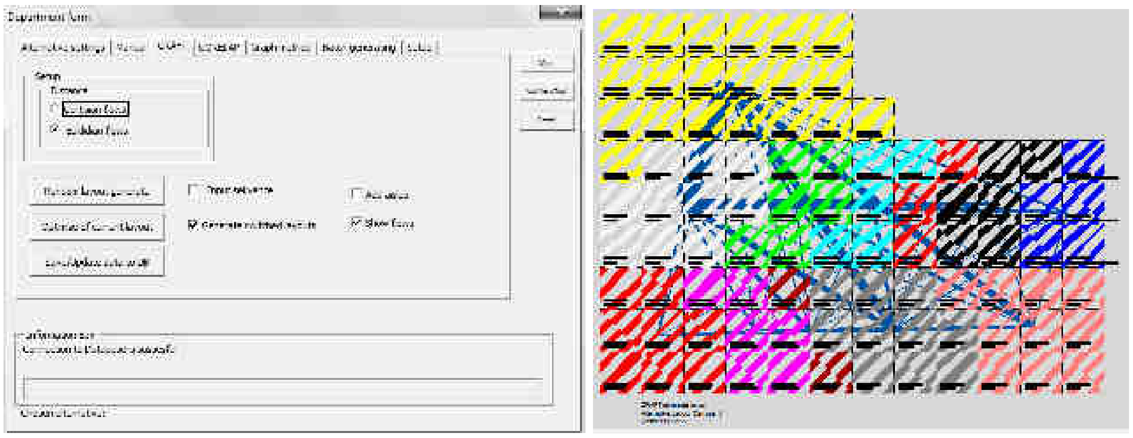


Figure 196 – CRAFT

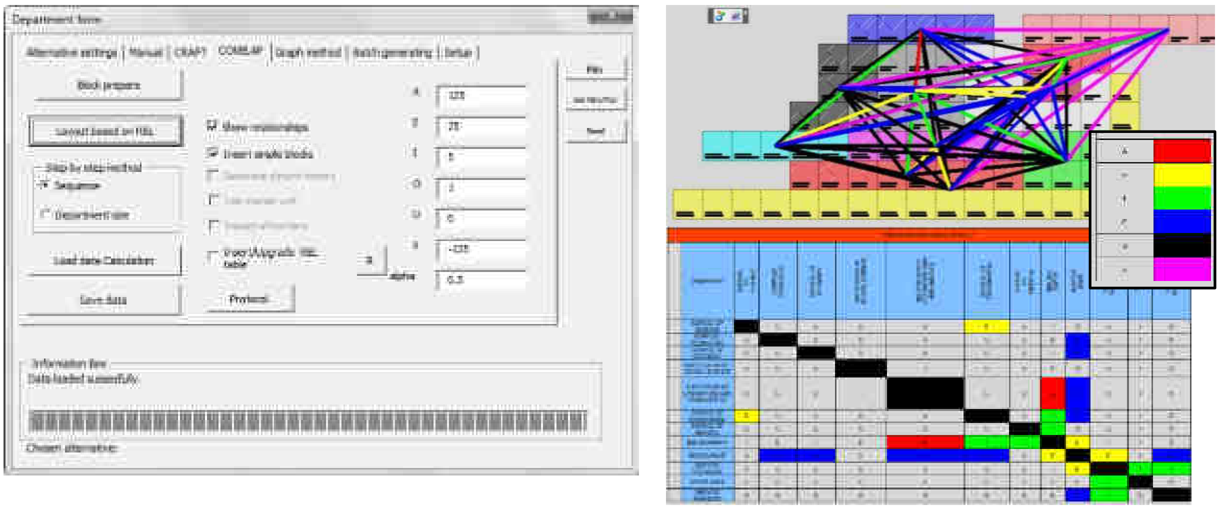


Figure 197 – CORELAP

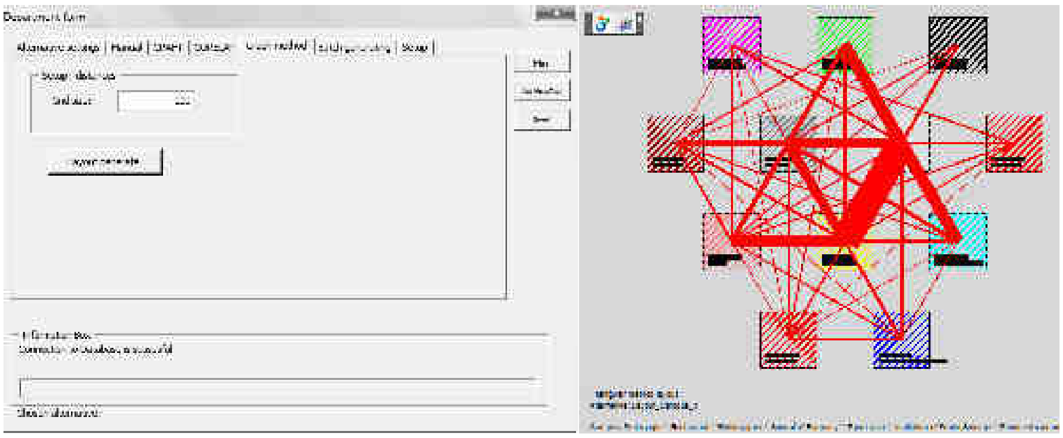


Figure 198 – Graph Theory Block Layout

A.4.7.3 CORELAP

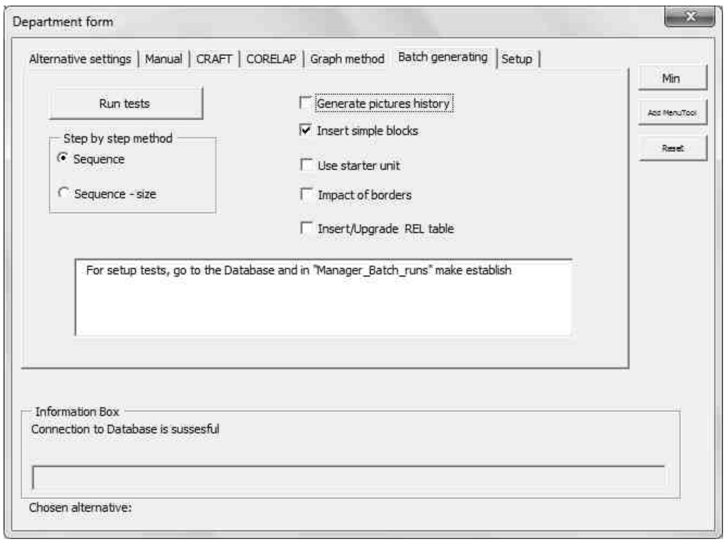
CORELAP form **IDS.CAD.FACTORY.CORELAP** (Figure 197) provides the generation of a selected alternative based on relationship charts with the given AEIOUX evaluation scale. Its implementation is described in Section 4.5.5. Two algorithm modifications are based on *sequence* or *department size*. The sequence modification generates units in cycles depending on departments; *department size* generates units in cycles based on the unit number.

Function *"Load_data_Calculation()"* evaluates the alternative by a score based on achieved relationships. It is helpful to see the improvements in the manual arrangement of layout. In order to see the achieved relationships, two options are available:

- relationships as coloured lines connecting centres of departments, each colour representing a relationship (**IDS.CAD.FACTOR.CORELAP**.***Show relationships*** = true)
- Relationship chart where items with coloured backgrounds mean achieved relationship (**IDS.CAD.FACTOR.CORELAP**.***Insert/Upgrade REL table*** = true)

A.4.7.4 Graph theory Block Layout

IDS.CAD.FACTORY.GRAPH_METHOD form provides the generation of the triangular grid with given dimensions (**IDS.CAD.FACTORY.GRAPH_METHOD**.***grid size***). Implementation is described in Section 4.5.6. This method does not consider department sizes and insert only one representing unit (see Figure 198).


Department form

Alternative settings | Manual | CRAFT | CORELAP | Graph method | Batch generating | Setup

Run tests

Step by step method
(●) Sequence
( ) Sequence - size

☐ Generate pictures history
☑ Insert simple blocks
☐ Use starter unit
☐ Impact of borders
☐ Insert/Upgrade REL table

For setup tests, go to the Database and in "Manager_Batch_runs" make establish

Information Box
Connection to Database is sussesful

Chosen alternative:

Figure 199 – Batch alternative generation

A.4.7.5 Other functionalities

IDS provides other functions that helps factory design. It is a **batch generation** of alternatives based on their configuration in the database table "Table_Run_Batch" (**IDS.DB.FACTORY_LAYOUT_DESIGN.BATCH_RUN**). It is possible to setup different input sequence, parameters (neighbour), and methods (CRAFT, CORELAP, Graph Theory Block Layouts) for all of alternatives. In Figure 199, the complete setup form is displayed. It is helpful for the **comparison of alternatives**. This comparison can be managed by the achieved scores based on material flow costs or relationship evaluation.

A.4.8 Deterministic models

As mentioned in the previous sections, it is possible to design deterministic and simplified models in IDS. Deterministic models do not consider random influences as in stochastic models. In IDS, the use is simplified due to the absence of simulation.

A.4.8.1 DB setup

The configuration of deterministic models (DM) is managed by the form **IDS.DB.DETERMINISTIC_MODELS** (see Figure 200). **IDS.DB.DETERMINISTIC_MODELS.INPUT_DATA** contains forms for Departments (or resources). Parts and Schedules/Input amount are the same as in the **IDS.DB.RESOURCES**. Form **IDS.DB.DETERMINISTIC_MODELS.DATA_PROCESSING** contains functions for filling other requirement data: places and routing information.

In DM, places are nested in departments and they are automatically filled by the function *"Generate_space_based_on_facility()"*. Data about part routing can be prepared by functions *"Preparing_Table_Storage_Machine()"* and *"Preparing_Table_Machine_Storage()"*. After that, space names have to be filled in forms **MANUAL_DATA_PREPARE_SM** and **MANUAL_DATA_PREPARE_MS**. Routing can also be given from CAD function – see next Section. Sequences of resources for a given part is setup in the table "Production line data" or from CAD. After that, the function *"Preparation_ Material flows()"* automatically fills information about material flows.

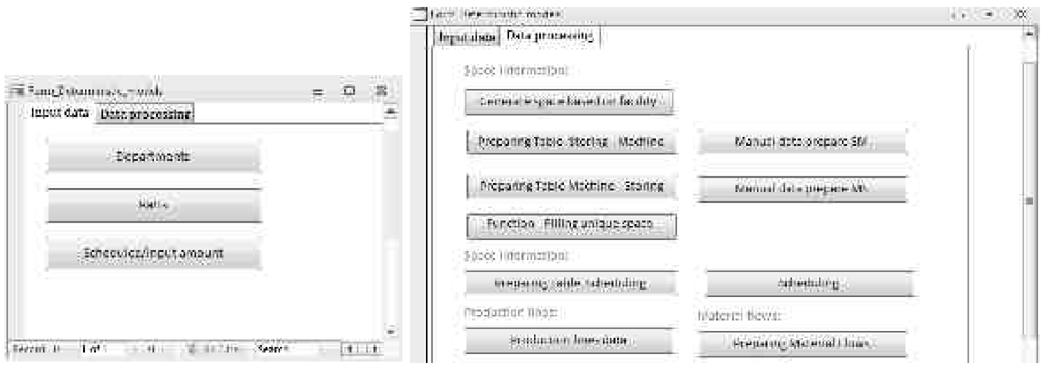


Figure 200 – Deterministic models

A.4.8.2 CAD setup

Working with DM is similar to working with the stochastic model. The differences are the special block used that represents the resource and buffer together: ***process_buffer*** block is described in Section 4.5.2 and an example is shown in Figure 201. The generation of material flows is done with the parameter **IDS.CAD.FACILITY.FACILITY_PLACEMENT.*Operation data inolved*** = false.

A.4.8.3 Loading operation sequences from a CAD layout

A sequence of operation can also be loaded from the layout that contains the resources. The principle is shown in Figure 201. The user connects basic points of resources to the polylines lying in the same layer as a part. After that, by the function *"Load_Material_Flows_from_layout()"* (form **IDS.CAD.FACILITY.SIMPLE_MODELS**), the operation sequences are recognised and saved into tables Table_Storage_Machine and Table_Machine_Storage. Then, these assistant polylines can be deleted and substituted by material flows.

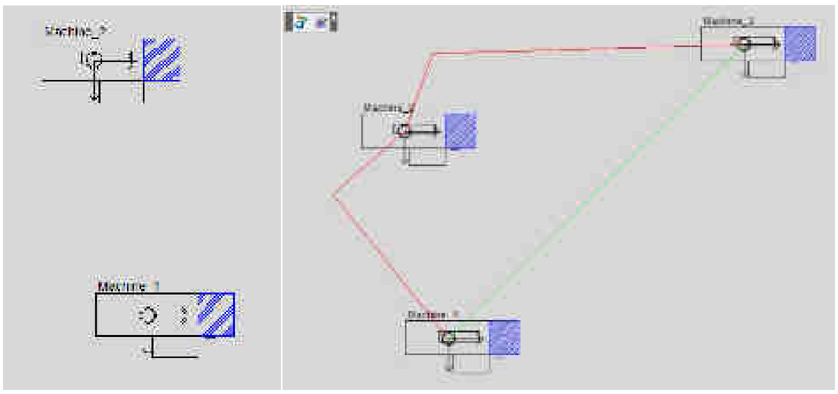


Figure 201 – Loading operation sequences from CAD layout

A.4.9 Layout patterns

IDS provides automatic generation of layout patterns based on the PQ analysis. There exist three different layout patterns designed by different approaches: **process layouts** (job-shop), **cellular layouts** and **production lines** for a specific product (see the Section 1.1.4). Other options for inserting blocks for a system design are: resources placed in the row (y coordinate is constant), in the column (x coordinate is constant) or in the diagonal.

The generation of layouts is based on the procedure *"Inserting of drawing blocks"* (see Section 4.5.2).

The following example shows differences in the design approach and the comparison of layout patterns. In this example, it is considered simple ***deterministic system***, without random influences and, therefore, there is no need to use simulation. It can also be used in cases where any data as operations are available. In this example, since the focus is on the principle of the generation of layout patterns, results are not important.

Input data processing

Input data is taken from the work (Murugan & Selladurai, 2005). Input data are list of 20 parts (**IDS.DB.RESOURCES.PARTS**), list of 15 machines (**IDS.DB.RESOURCES.FACILITIES**), Product-Machine matrix (**IDS.DB.PRODUCTION_ANALYSE.TABLE_MACHINE_PART**), production volumes

(**IDS.DB.DETERMINISTIC_MODELS.SCHEDULING**) and part routing through the machines (**IDS.DB.DETERMINISTIC_MODELS.PRODUCTION LINES**).

Pattern generation

The generation of layout patterns is processed by a configuration of **IDS.CAD.FACILITY.FACILITY_PLACEMENT** form (see Figure 202). The layout pattern can be chosen as basic point coordination of a layout or gaps between inserted blocks. After that, the layout can be generated by function *"Generate layout()"*.

<u>Setup of the IDS properties:</u>

IDS.CAD.FACILITY.FACILITY_PLACEMENT.*<u>Operation data involved</u>* = false (only deterministic data)

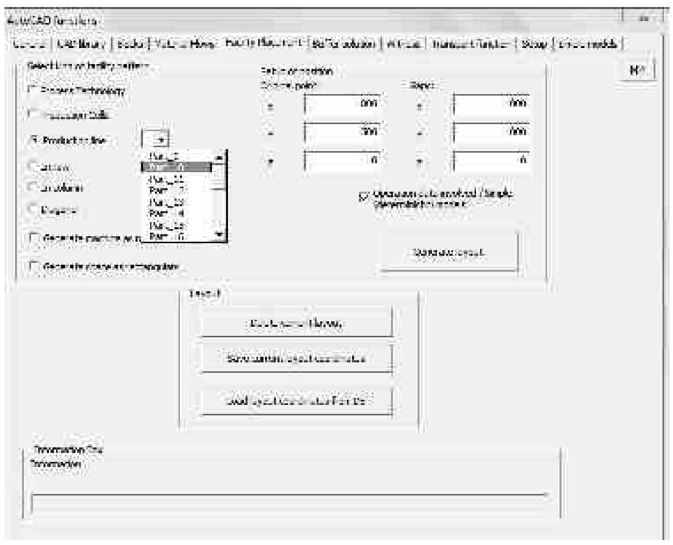


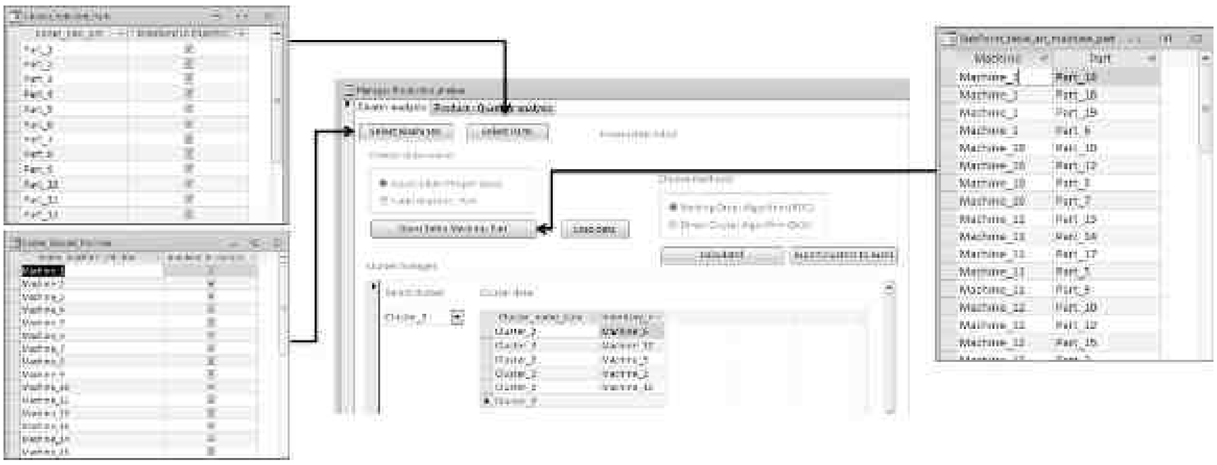
Figure 202 - CAD form: Facility placement

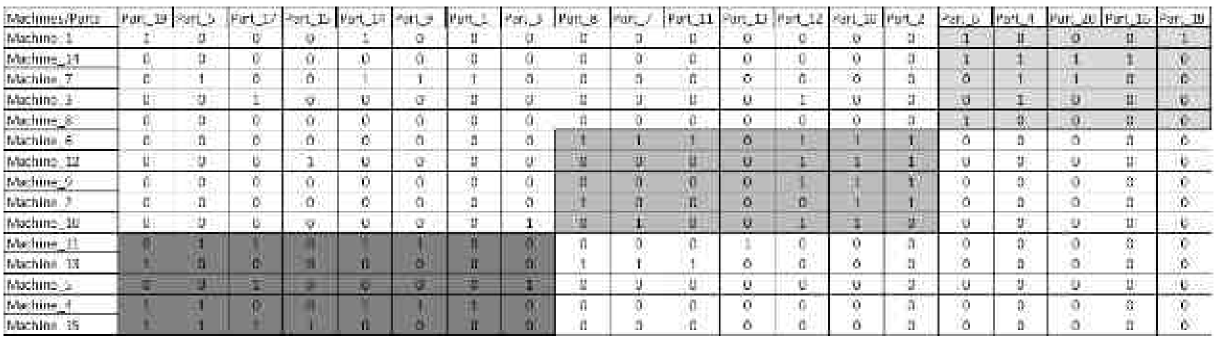
A.4.9.1 Cellular layouts

This type of layout, also called as "Group technology" layout, is based on a cluster analysis of input data. Clusters are established as groups of machines producing a specific set of products, as shown in Figure 203. It is done by functions in the form **IDS.DB.PRODUCTION_ANALYSE.CLUSTER_ANYLYSIS.**

It is possible to select Parts and Machines that are involved in cluster analysis by functions *"Select Machines"* and *"Select Parts"*. Two possibilities for input data set can be used: data used in the simulation project or data from Table Machine-part for deterministic model. The first data set is based on data from a simulation model. This data are automatically prepared from entities routing and storing information (see Section A.4.2). The second possible input data set is just a table that contains machines and parts (**IDS.DB.PRODUCTION_ANALYSE.TABLE_MACHINE_PART**).

For cluster design in IDS, two algorithms are implemented: ROC and DCA. After "*Loading*" data and "*Calculation*", results are "*Exported*" in the excel format "csv". In MS Excel, clusters are easily designed, as shown in the bottom image of the Figure 203. Clusters data are then recorded into DB (i.e. list of grouped machines in the appropriate cluster). The rest of the machines, which are not included in any cluster, are placed in the layout as exceptions in a special last group.





Machines/Parts	Part_19	Part_5	Part_17	Part_15	Part_14	Part_9	Part_1	Part_3	Part_8	Part_7	Part_11	Part_13	Part_12	Part_10	Part_2	Part_6	Part_4	Part_20	Part_16	Part_18
Machine_1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
Machine_14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0
Machine_7	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	1	1	0	0
Machine_3	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0
Machine_8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Machine_6	0	0	0	0	0	0	0	0	1	1	1	0	1	1	1	0	0	0	0	0
Machine_12	0	0	0	1	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0
Machine_9	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0
Machine_2	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0
Machine_10	0	0	0	0	0	0	0	1	1	1	0	0	1	1	1	0	0	0	0	0
Machine_11	0	1	1	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
Machine_13	1	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
Machine_5	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Machine_4	1	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Machine_15	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 203 - Process of cluster design

In Figure 204, cellular layouts are automatically generated. The upper image contains a drawing block of machines in the clusters and material flow of the selected part (yellow colour). The figure below displays the overall material flows. The Figure 205 shows results and analysis of material flows from the form **IDS.DB.MATERIAL_FLOW_ANALYSE**.

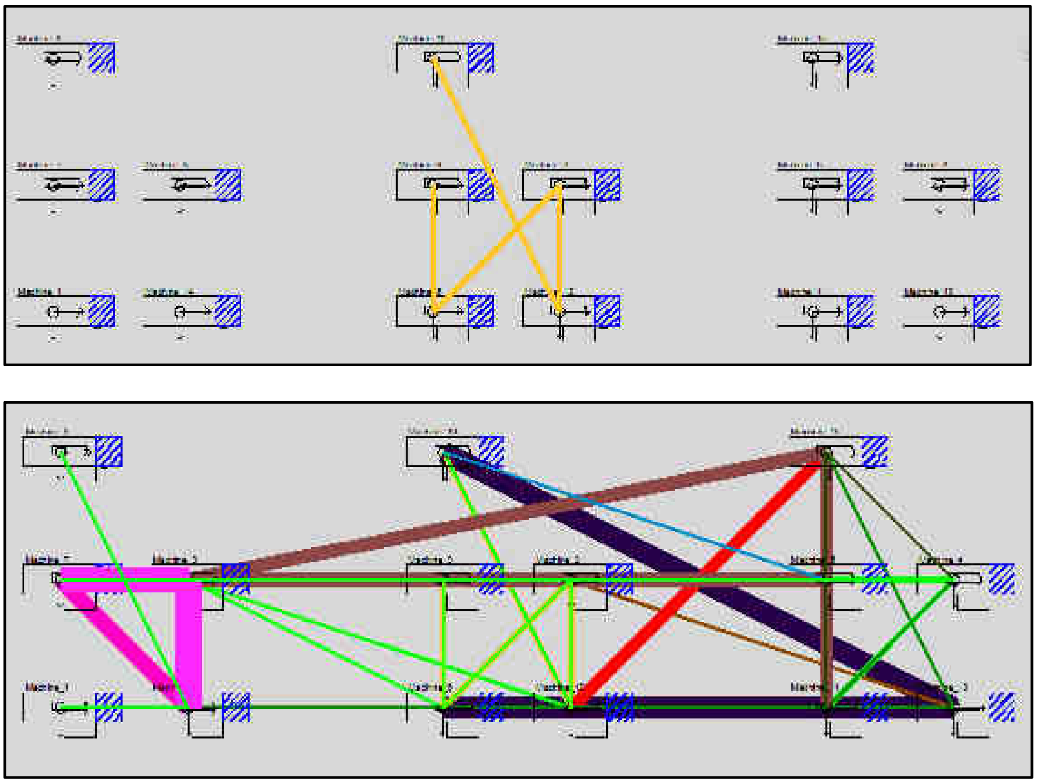
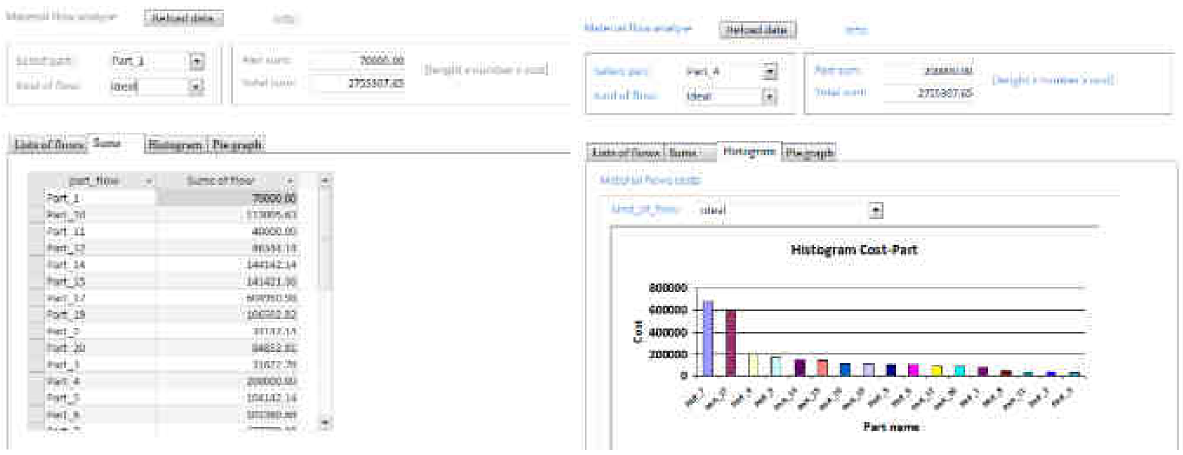


Figure 204 - Layouts of clusters



Material flow analyser [Reload data] Info

Select part	Part_1
Kind of flows	ideal
Part sum	70000.00
Total sum	2725307.65

[weight x number x cost]

Lists of flows | Sums | Histogram | Pie graph

part_flow	Sums of flow
Part_1	70000.00
Part_10	113895.63
Part_11	40000.00
Part_12	86334.18
Part_14	144142.14
Part_15	141421.36
Part_17	609760.56
Part_19	106062.82
Part_2	31112.15
Part_20	84852.81
Part_3	31622.78
Part_4	200000.00
Part_5	104142.14
Part_6	102380.89

Material flow analyser [Reload data] Info

Select part	Part_4
Kind of flows	ideal
Part sum	200000.00
Total sum	2725307.65

[weight x number x cost]

Lists of flows | Sums | Histogram | Pie graph

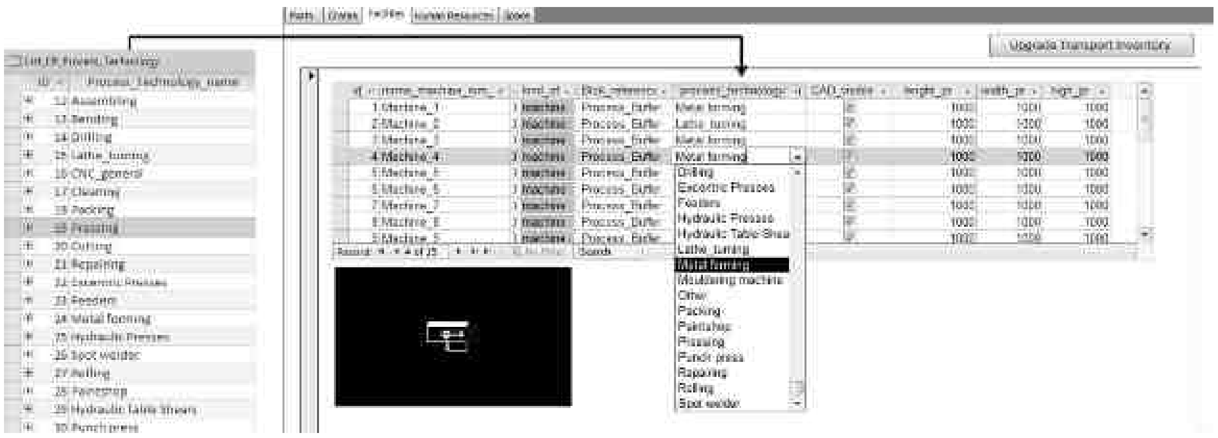
Material flows costs

Kind_of_flow: ideal

Figure 205 - Results of material flows in the Cellular layouts

A.4.9.2 Process layouts

Process layouts, also called job-shop, are layouts where machines with the same kind of technology are grouped together. In Figure 206, there is the **IDS.DB.Resource.Facility** form with a list of machines and a table with technology items.



List_Of_Process_Technology

ID	Process_Technology_name
12	Assembling
13	Bending
14	Drilling
15	Lathe_turning
16	CNC_general
17	Cleaning
18	Packing
19	Pressing
20	Cutting
21	Repairing
22	Excentric Presses
23	Feeders
24	Metal forming
25	Hydraulic Presses
26	Spot welder
27	Rolling
28	Paintshop
29	Hydraulic table Shears
30	Punch press

id	name_machine_form_	kind_of	Block_reference	process_technology	CAD_visible	lenght_pr	width_pr	high_pr
1	Machine_1	machine	Process_Buffer	Metal forming	☑	1000	1000	1000
2	Machine_2	machine	Process_Buffer	Lathe_turning	☑	1000	1000	1000
3	Machine_3	machine	Process_Buffer	Metal forming	☑	1000	1000	1000
4	Machine_4	machine	Process_Buffer	Metal forming	☑	1000	1000	1000
5	Machine_5	machine	Process_Buffer	[illegible]	☑	1000	1000	1000
6	Machine_6	machine	Process_Buffer	[illegible]	☑	1000	1000	1000
7	Machine_7	machine	Process_Buffer	[illegible]	☑	1000	1000	1000
8	Machine_8	machine	Process_Buffer	[illegible]	☑	1000	1000	1000
9	Machine_9	machine	Process_Buffer	[illegible]	☑	1000	1000	1000

Record: 4 of 15 | No Filter | Search

Drilling
Excentric Presses
Feeders
Hydraulic Presses
Hydraulic Table Shea
Lathe_turning
Metal forming
Moulding machine
Other
Packing
Paintshop
Pressing
Punch press
Repairing
Rolling
Spot welder

Figure 206 - Process layouts assigning

In this example, three different technologies are used: Lathes, Metal forming and Pressing machines as it is shown in the genereted layouts in Figure 207. There are three departments focused on different technologies. The image below displays the overall material flows in the system where the yellow colour ilustrates the material flow of a chosen part.

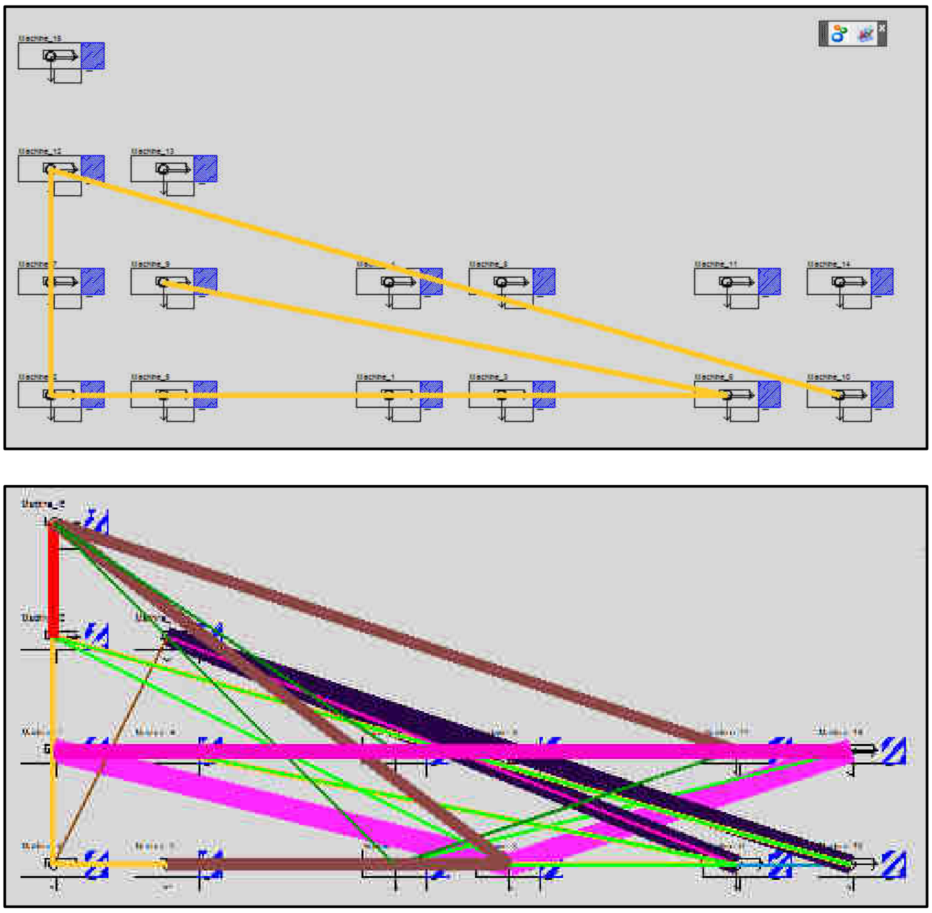
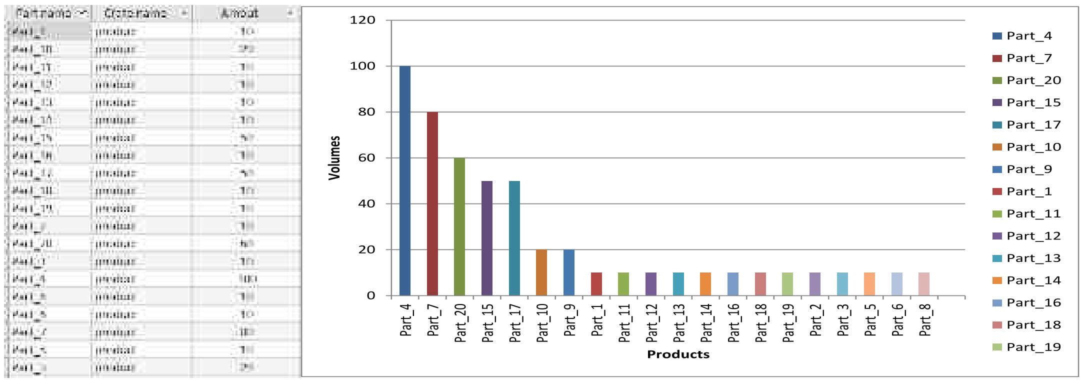


Figure 207 - Process layouts

A.4.9.3 Production lines layouts

Production lines layouts, also called *flow-shop*, are focused on the specific product with the highest volumes. These products can be established based on a P-Q analysis, as mentioned in the Section 2.1.1.

The production volumes in this example is shown in Figure 208, as well as the PQ analysis that shows parts in decreasing order, based on their volumes. This form is **IDS.DB.PRODUCTION_ANALYSIS.PRODUCT-QUANTITY_ANALYSIS.**



Partname	[illegible]	Amount
Part_1	product	10
Part_10	product	20
Part_11	product	10
Part_12	product	10
Part_13	product	10
Part_14	product	10
Part_15	product	50
Part_16	product	10
Part_17	product	50
Part_18	product	10
Part_19	product	10
Part_2	product	10
Part_20	product	60
Part_3	product	10
Part_4	product	100
Part_5	product	10
Part_6	product	10
Part_7	product	80
Part_8	product	10
Part_9	product	20

Figure 208 - PQ analysis

IDS supports the generation of production line for a selected part, as shown in Figure 209.

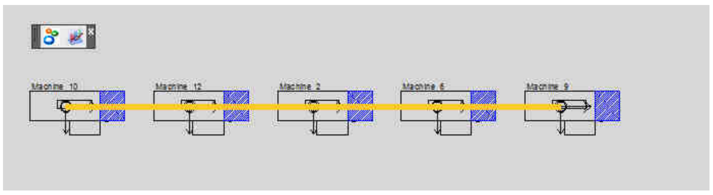


Figure 209 - Production line

A.4.10 Example: Forklift transport system

This example explains a principle of taxi transportation system design and the integration between CAD data and simulation experiments with feedbacks. It is based on the concept described in Section 4.7.1.

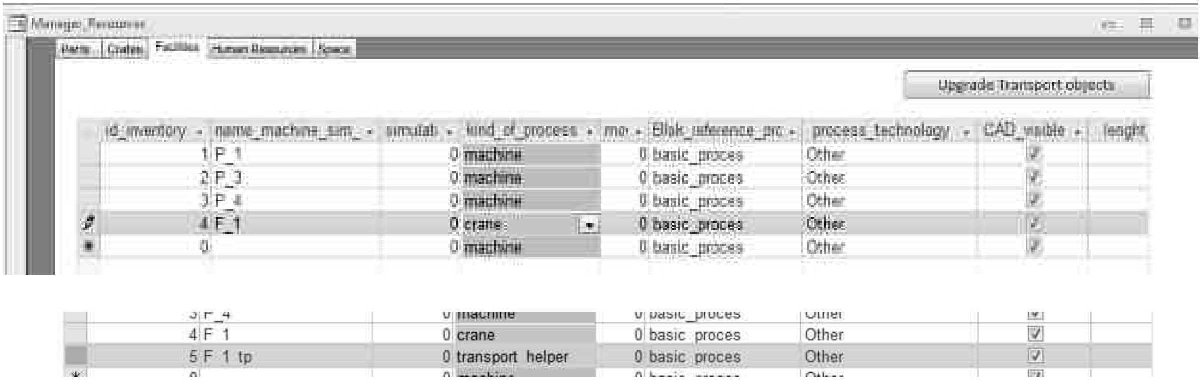
A.4.10.1 Model description

This model describes a system of two forklifts transporting items from *"Start position"* into *"Destination position"*. Transportation times are derived from the forklift's speeds and the length of transportation routes designed in the CAD layout. Transported flows can be automatically generated by the function *"Shortest path"*, according to the shortest path found between two stations in the **transportation routing network** (see details in Section 4.7.3). This example has the following phases:

- Preparing data
- Transportation material flows generation
- Data feedback controlling

A.4.10.2 Preparing data and system setup

In this phase, the model is configured and the required data are inserted in DB. Transporters have to be distinguished by **IDS.DB.RESOURCES.FACILITIES.*kind_of_process*** = "crane".
Based on Section 4.7.1, some specific objects are used as a transportation helper entity, etc. IDS provides the function for a full-automatic filling of these data: **IDS.DB.RESOURCES.FACILITIES. *Upgrade_Transport_objects().*** The following data are added and checked: ListOfInventory table (add [Transporter][_tp] as transportation helper resource), ListOfParts table (add [Transporter][_dm] as transportation helper entity), ListOfSpace table (add [Transporter][_plc] as the capacity of transporter) and tables Input and Output (filling values of order_in (=2) and order_out (=1)).



	id_inventory	name_machine_sim_	simulati	kind_of_process	mo	Blok_reference_pro	process_technology	CAD_visible	lenght
	1	P_1	0	machine	0	basic_proces	Other	☑	
	2	P_3	0	machine	0	basic_proces	Other	☑	
	3	P_4	0	machine	0	basic_proces	Other	☑	
✎	4	F_1	0	crane	0	basic_proces	Other	☑	
*	0		0	machine	0	basic_proces	Other	☑	

	4	F_1	0	crane	0	basic_proces	Other	☑
	5	F_1_tp	0	transport_helper	0	basic_proces	Other	☑

Figure 210 - Transportation

<u>Setup of the IDS properties:</u>

Operation:

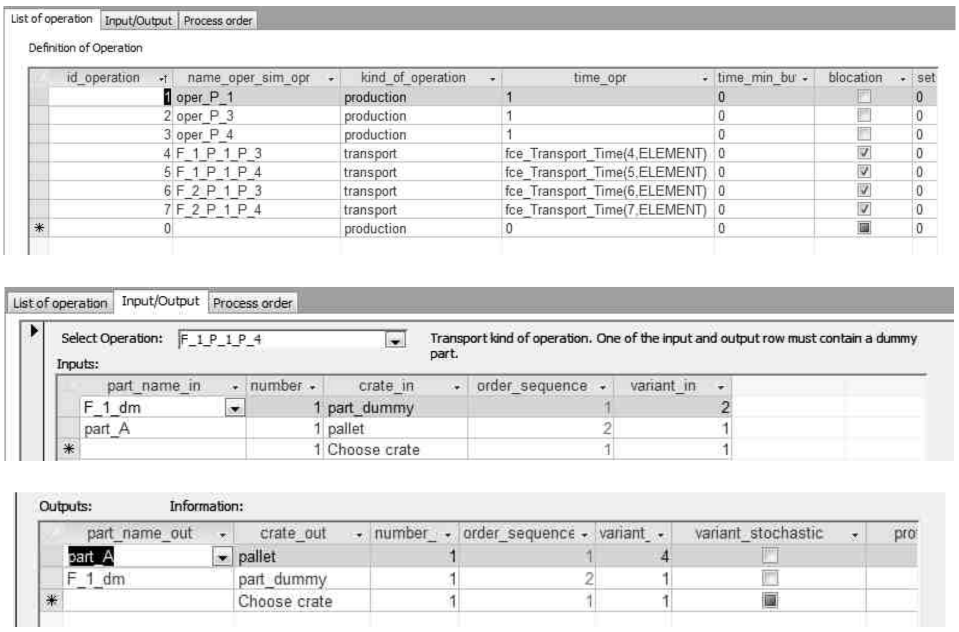
IDS.DB.OPERATION_SETUP.LIST_OF_OPERATION.*kind_of_operation* = *"transport"*

IDS.DB.OPERATION_SETUP.LIST_OF_OPERATION.*time_oper* = *„fce_Transport_Time(ID_operation, ELEMENT)"*

IDS.DB.OPERATION_SETUP.LIST_OF_OPERATION.*blocation* = *true*

IDS.DB.OPERATION_SETUP.INPUT: one inputting part is a transportation helper entity (*"dummy"*) with *"router = 2"*

IDS.DB.OPERATION_SETUP.OUTPUT: one outgoing part is a transportation helper entity (*"dummy"*) with *"router = 1"*


List of operation | Input/Output | Process order

Definition of Operation

id_operation	name_oper_sim_opr	kind_of_operation	time_opr	time_min_bu	blocation	set
1	oper_P_1	production	1	0	☐	0
2	oper_P_3	production	1	0	☐	0
3	oper_P_4	production	1	0	☐	0
4	F_1_P_1_P_3	transport	fce_Transport_Time(4,ELEMENT)	0	☑	0
5	F_1_P_1_P_4	transport	fce_Transport_Time(5,ELEMENT)	0	☑	0
6	F_2_P_1_P_3	transport	fce_Transport_Time(6,ELEMENT)	0	☑	0
7	F_2_P_1_P_4	transport	fce_Transport_Time(7,ELEMENT)	0	☑	0
0		production	0	0	■	0

List of operation | Input/Output | Process order

Select Operation: F_1_P_1_P_4

Transport kind of operation. One of the input and output row must contain a dummy part.

Inputs:

part_name_in	number	crate_in	order_sequence	variant_in
F_1_dm	1	part_dummy	1	2
part_A	1	pallet	2	1
	1	Choose crate	1	1

Outputs: Information:

part_name_out	crate_out	number_	order_sequence	variant_	variant_stochastic	pro
part_A	pallet	1	1	4	☐	
F_1_dm	part_dummy	1	2	1	☐	
	Choose crate	1	1	1	■	

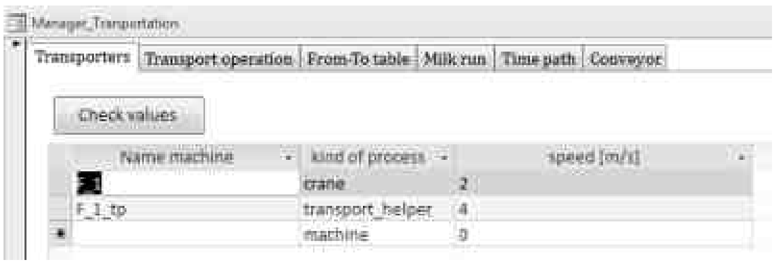
Figure 211 - Transportation operation setup

Transportation data:

Form **IDS.DB.TRANSPORTATION** involves the setup of the transportation as:

IDS.DB.TRANSPORTATION.TRANSPORTERS.*<u>speed</u>* = value (unit - meter per second)

It is possible to setup different speeds for an empty (transporter helper) and a loaded transporter.


Manager_Transportation

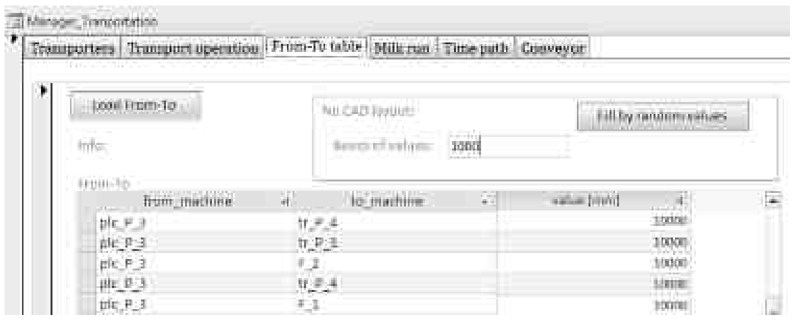
Transporters | Transport operation | From-To table | Milk run | Time path | Conveyor

Check values

Name machine	kind of process	speed [m/s]
[illegible]	crane	2
F_1_tp	transport_helper	4
	machine	0

Figure 212 - Transporters properties

The names of buffers "from_position" and "to_position" are automatically filled based on data from "Table_Machine_Storage" and "Table_Storage_Machine ".


Manager_Transportation

Transporters | Transport operation | From-To table | Milk run | Time path | Conveyor

Load From-To

Info:

No CAD layout:

Fill by random values

Range of values: 1000

From-To

from_machine	to_machine	value [mm]
plc_P_3	tr_P_4	10000
plc_P_3	tr_P_3	10000
plc_P_3	F_2	10000
plc_P_3	tr_P_4	10000
plc_P_3	F_1	10000

Figure 213 - Transportation – *from-to* distances

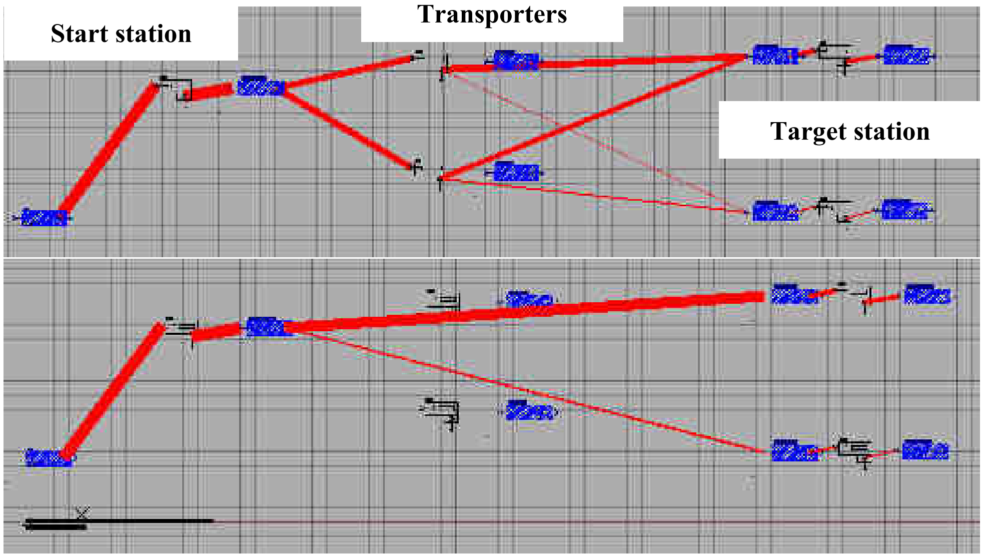


Figure 214 - Transportation: ideal flows

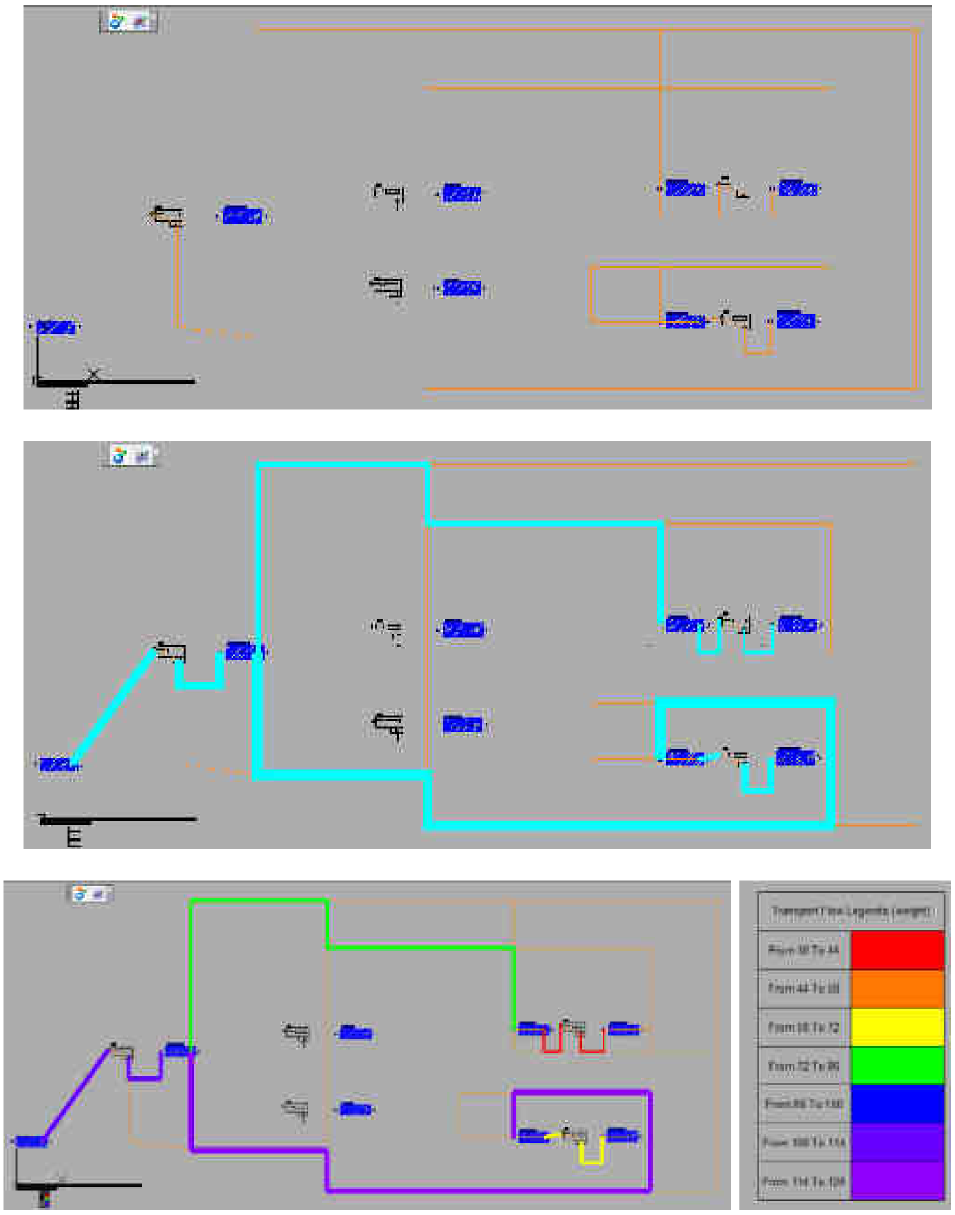


Figure 215 - Transportation network and material flows generated by IDS

Transportation distances can be filled manually, by random values or according to the length of material flows taken from the layout. The two first operations can be used in case of data absence.

Based on these data, the **simulation model** can be generated and experiment runs to receive values of material flows between resources and buffers (see Section A.4.4).

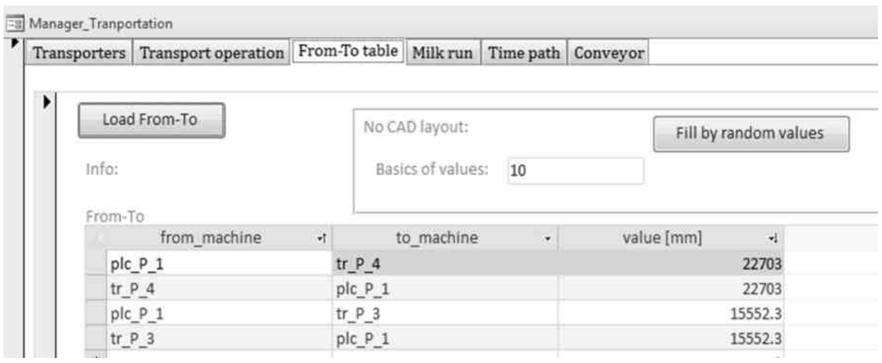
A.4.10.3 Transportation material flows generation

The layout can be generated by the involved functions (see Section A.4.8) as well as by the material flows. By the parameter **IDS.CAD.FACILITY.MATERIAL_FLOWS.*transport_flow*** = true, flows are generated directly between loading and the target station, which makes it possible to skip the transporter resource. The differences are shown in Figure 214, where the first possibility can be helpful to see the full routing map of entities.

Another option would be to generate flows based on the transportation network (parameter **IDS.CAD.FACILITY.MATERIAL_FLOWS.**_**Shortest_Path_Generated_Flows**_ = true). In Figure 215, there is the **transportation routing network** (orange colour) that represents aisles for transporter movements. The middle image shows material flows for the selected entity and the image below shows the overall transportation intensity, with a legend where each colour specifies different flow intensity for a given segment of network.

A.4.10.4 Feedback control

After the design of material flows, it is possible to load their lengths to DB and use them in simulation model as upgraded transportation times, by using the function **IDS.DB.TRANSPORTATION.FROM_TO_TABLE.*Load_From_to()***. Figure 216 shows a snapshot of upgraded values that can be used for a new simulation running.



Basics of values: 10

from_machine	to_machine	value [mm]
plc_P_1	tr_P_4	22703
tr_P_4	plc_P_1	22703
plc_P_1	tr_P_3	15552.3
tr_P_3	plc_P_1	15552.3

Figure 216 - Distances based on material flows

A.4.11 Example: Milk-run delivering system

The IDS system provides a set of functions supporting the design of milk-run (MR) delivering and transportation system. The principle of MR is described in Section 4.7.2.

A.4.11.1 Model description

In this example, there is a system of two MRs that supplies six stations. In Figure 217, there is a schema of this system. These stations consume products taken from boxes (transported unit). Full and empty boxes are delivered and picked-up by MR based on the kanban system, i.e. change empty boxes by full in the warehouse. The MR system can be defined as a transportation cycle where all stations are visited. This system consists of sub-cycles where the amount of boxes is filled.

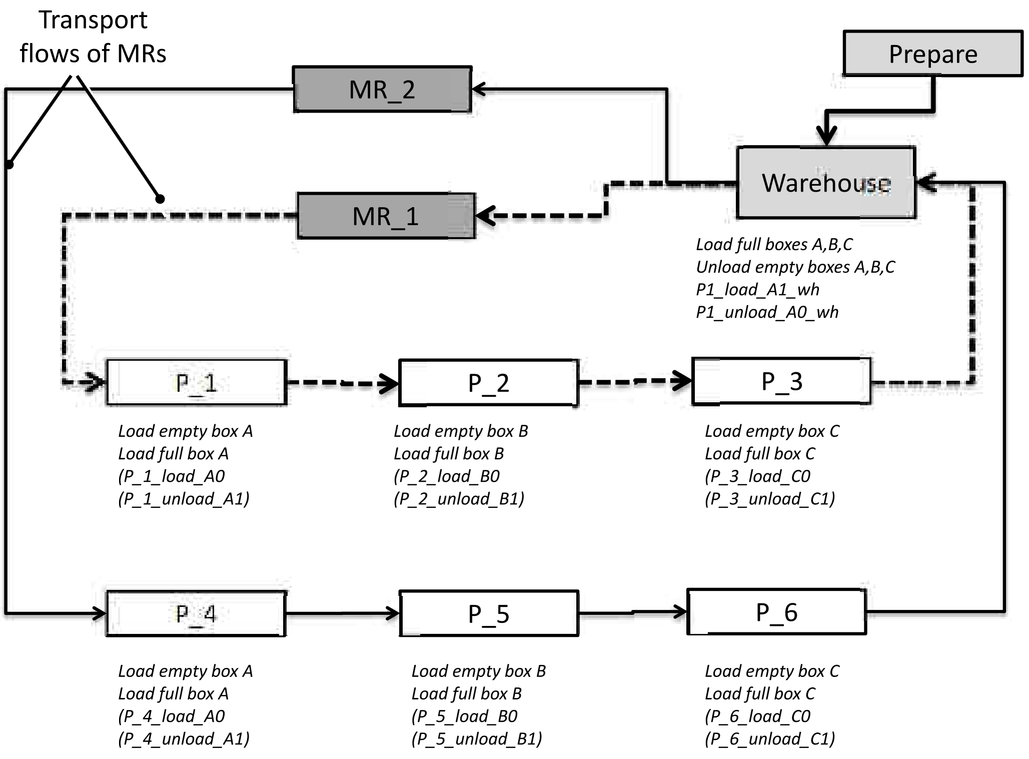


Figure 217 – Schema of the MR system

A.4.11.2 Input data and system setup

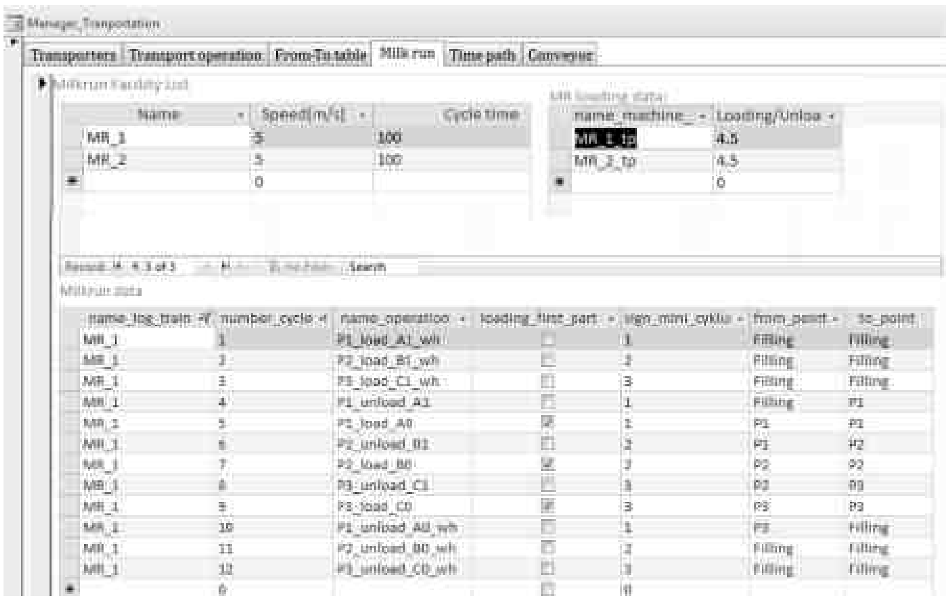
Resources

Each MR is a transportation kind of resource, as defined in the form: **IDS.DB.RESOURCE.FACILITIES.***kind of transport* = “milk_run”. MR also need a transportation helper entity and other objects. All these objects are automatically filled and upgraded by the function **IDS.DB.RESOURCES.FACILITIES.** ***Upgrade Transport objects()***.

Operations

The MR transportation system consists of a sequence of operations defined by a movement of MR in the layout and delivering items to the stations. The definiton of **supplying cycle** for each MR states that each step (operation) is asssigned by a sequence number. In Figure 218, there is the form **IDS.DB.TRANSPORTAION.MILK_RUN** for the setup of the MR properties as speed, the MR cycle times and loading/uploading times. If this cycle time is too short than the required time for delivering and transportation, an error message appears ("*negative setup time*") during the WITNESS simulation running.

The sequence of operation is defined by the property *"number_cycle"*. Sub-cycles, describing exchanging boxes (operations: loading empty boxes in the station, unloading empty boxes in the warehouse, loading full boxes in the warehouse and unloading full boxes in the station), are enumerated in the property *"sign_mini_cyclus"*. A specific number is for one part. This cycle must be also defined by a starting operation, that is loading empty boxes, signed by *"loading_first_part"* = true. This table also involves information about movement during each operation: *"from_point"* to *"to_point"*. Distances between these points are defined in the **IDS.DB.TRANSPORTAION.FROM_TO** – see Figure 216.


Manager_Transportation

Transporters | Transport operation | From-To table | Milk run | Time path | Conveyor

Milkrun Facility List

Name	Speed[m/s]	Cycle time
MR_1	5	100
MR_2	5	100
	0	

MR loading data

name_machine_	Loading/Unloa
MR_1_tp	4.5
MR_2_tp	4.5
	0

Record: 3 of 3 | No Filter | Search

Milkrun data

name_log_train	number_cycle	name_operation	loading_first_part	sign_mini_cyklu	from_point	to_point
MR_1	1	P1_load_A1_wh	☐	1	Filling	Filling
MR_1	2	P2_load_B1_wh	☐	2	Filling	Filling
MR_1	3	P3_load_C1_wh	☐	3	Filling	Filling
MR_1	4	P1_unload_A1	☐	1	Filling	P1
MR_1	5	P1_load_A0	☑	1	P1	P1
MR_1	6	P2_unload_B1	☐	2	P1	P2
MR_1	7	P2_load_B0	☑	2	P2	P2
MR_1	8	P3_unload_C1	☐	3	P2	P3
MR_1	9	P3_load_C0	☑	3	P3	P3
MR_1	10	P1_unload_A0_wh	☐	1	P3	Filling
MR_1	11	P2_unload_B0_wh	☐	2	Filling	Filling
MR_1	12	P3_unload_C0_wh	☐	3	Filling	Filling
	0		☐	0		

Figure 218 –Setup of the MR system

All these operations must be signed as "transport" in the **IDS.DB.OPERATION_SETUP**.***<u>kind of operation</u>*** = "transport".

A.4.11.3 Design of routes in the layout

The layout for this system can be generated by the function in the form **IDS.CAD.FACILITY.FACILITY_PLACEMENT**, described in Section A.4.8. By the functions in the form **IDS.CAD.FACILITY.MATERIAL_FLOWS** (use description is in Section A.4.4, it is possible to generate material flows. Snapshots are in Figure 219, where the first image shows material flows for all parts in the system, distinguished by different colours; the second image displays material flows based on transportation units - red colour for full, yellow for empty boxes.

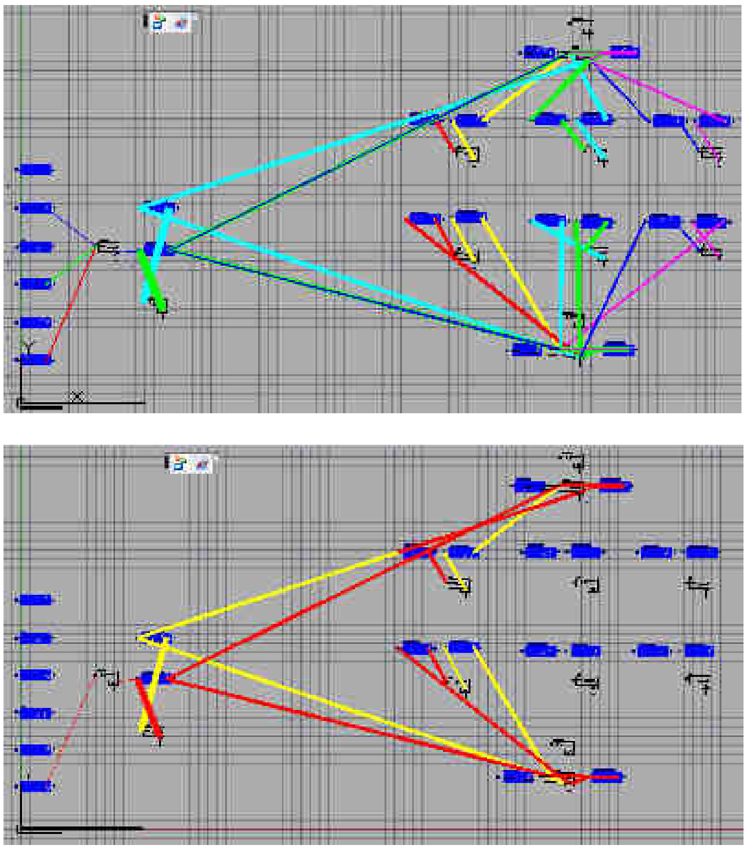


Figure 219 –Transportation flows

A.4.11.4 Simulation model

Simulation model is generated by the function included in the form **IDS.DB.GENERATION_SIM_MODELS** (Section A.4.3). In Figure 220, there is a generated model; the same model is also in Figure 185, with objects positioned according to the layout (use of WITNESS coordination function).

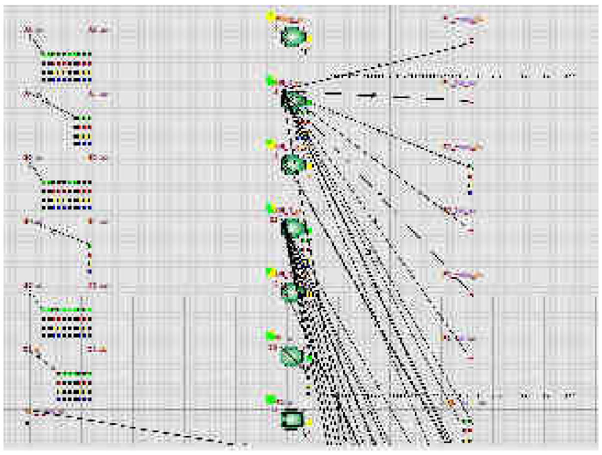
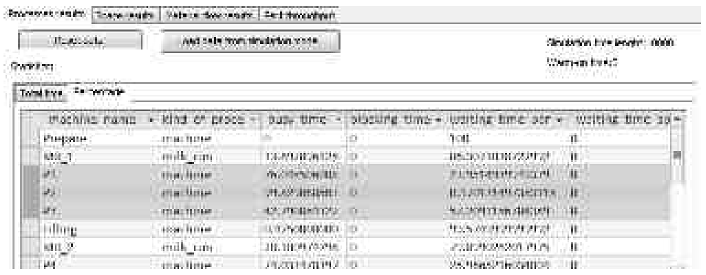


Figure 220 – WITNESS Simulation model

Statistics' results of the simulation experiment are in Figure 221. As results shown, this transportation system is not very well configurated because of resources P1, P3, P4 (stations), etc. that have a quite high percentage of waiting time. It means that there is not enough full boxes for making production operations. The system's performance can be improved by adding more boxes to the system or decreasing the milk-run cycle time. The configuration of the milk-run system is described in the practical example in Section 5.2.



machine name	kind of proce	busy time	blocking time	waiting time for	waiting time to
Prepare	machine	[illegible]	0	100	0
MR_1	milk_run	[illegible]	0	[illegible]	0
P1	machine	[illegible]	0	[illegible]	0
P2	machine	[illegible]	0	[illegible]	0
P3	machine	[illegible]	0	[illegible]	0
Filling	machine	[illegible]	0	[illegible]	0
MR_2	milk_run	[illegible]	0	[illegible]	0
P4	machine	[illegible]	0	[illegible]	0

Figure 221 – Results of the simulation experiment

A.4.12 Example: Supplying a production line

This illustrative example demonstrates a simple internal logistic system: supplying production facility by items involving unloading pallets from arrival truck, storing pallets and supplying production facility. In the previous sections, functions are described in details, as the generation of simulation models and layouts patterns, the design of material flows, buffer sizing, etc. In Figure 222, there is a schema of this system.

(1) *"Truck source":* Trucks arrive according to a schedule. It is possible to arrange the best plan for arrivals in order to achieve a safety storage level to avoid lack of parts for production)

(2) After going through the *"Check gate"*, the truck unloads cargo (full pallets) and waits for empty pallets to pick them up.

(3) Unloaded pallets are transported by forklift to the "*Storage".*

(4) From storage, pallets are transported near to the assembly workstation (*"Place for pallets")*

(5) where they are unpacked into boxes (*"Unpacking pallet").*

(6) Boxes are unpacked into parts. (*"Unpacking boxes")*

(7) Empty boxes are thrown away to the *"Waste".*

(8) Parts are assembled together in the *"Assembly workstation".*

(9) Empty pallets are transported from *"Place for pallets" (*after operation *Unpacking pallet"*) by forklift back to the empty truck that is nearby in order to be loaded and transported away through "*Check gate*".

A.4.12.1 Input data processing in the database

Input forms for this example are shown in the previous sections: parts (Figure 173), crates (Figure 174), resources (Figure 169), space (Figure 171) and operations (Figure 179).

Weights of entities loaded on the crates as well as transportation costs are given by the form

IDS.DB.WEIGHT_PARTS. Values are pre-filled by function *"Upgrade_Weigth()"*.

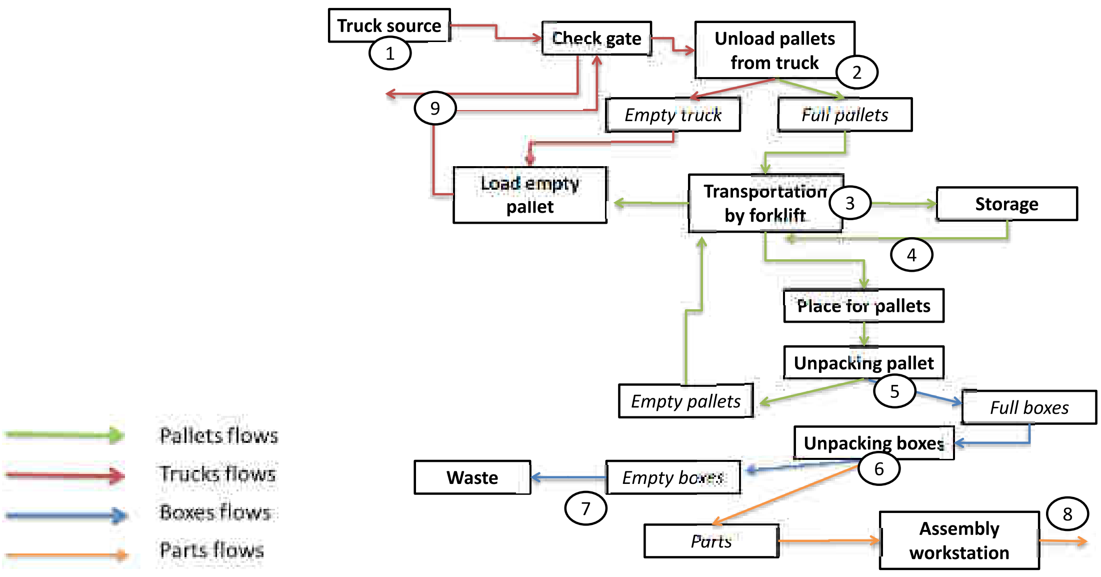


Figure 222 - Schema of internal logistic system

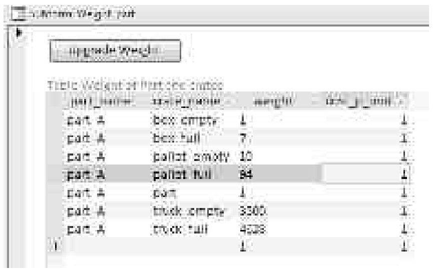


Table Weight of Part and crates

part_name	state_name	weight	[illegible]
part A	box empty	1	1
part A	box full	7	1
part A	pallet empty	10	1
part A	pallet full	84	1
part A	part	1	1
part A	truck empty	3300	1
part A	truck full	4528	1
		1	1

Figure 223 - Weight of parts and crates

Principles of the use of basic functions are described in the previous examples:

Unpack operation

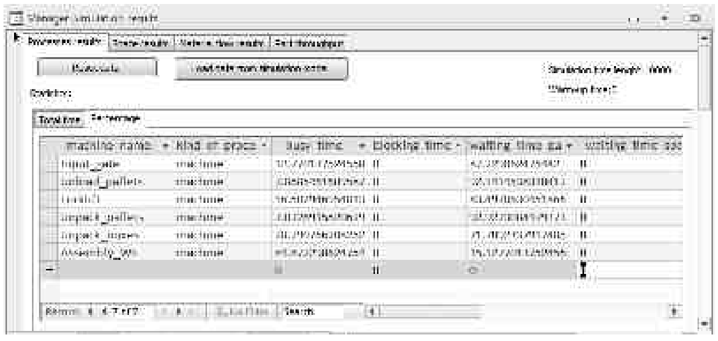
The principle of the unpacking operation is described in Section A.4.2, briefly: pallet, as a carrier of the load, can only be transported after all objects are removed from it.

Transportation

In this example, a transporter (forklift) is used to move pallets in the system. Transportation times are established as average values. It is a simplification in the system description. If it is required, it is possible to use **transportation function** that calculates the exact transportation time based on the length of material flows and vehicle speed and based on the current position in the system, as described in Section A.4.5.

A.4.12.2 Generated simulation model

Simulation model is generated by a function in the form **IDS.DB.GENERATION_SIM_MODELS**. A snapshot of a running model is in Figure 183. Results from the simulation experiment are in the form **IDS.DB.SIMULATION_RESULTS.PROCESS_RESULTS** in Figure 224.



machine name	kind of proce	busy time	blocking time	waiting time pa	waiting time sta
input_gate	machine	[illegible]	0	[illegible]	0
unload_pallets	machine	[illegible]	0	[illegible]	0
[illegible]	machine	[illegible]	0	[illegible]	0
unpack_pallets	machine	[illegible]	0	[illegible]	0
unpack_boxes	machine	[illegible]	0	[illegible]	0
Assembly_WS	machine	[illegible]	0	[illegible]	0

Figure 224 - Simulation results

A.4.12.3 AutoCAD layout

The design of layouts can be divided into several phases, from the ideal to constrained (realistic) layout. The **ideal layout** can be generated by functions involved in the form **IDS.CAD.FACILITY.FACILITY_PLACEMENT** which are described in Section A.4.8. Objects (resources and buffers) are generated by selected layout patterns (cells, production lines, etc.) or by other regular configurations into the grid. Actual gaps between object are not taken into consideration, neither their dimension. In this type of layouts, simple 2D rectangular blocks are used (black rectangular for resources, blue for buffers), as shown in Figure 225.

The next step in layout design is to exchange simple drawing blocks by **actual shapes** of appropriate machines, transporters, shelves, workers, etc. Changing blocks in control through **IDS.DB.RESOURCES** form, where the user can select a drawing block from the **3D library**, as described in Section A.4.1. In some cases, resources are only virtual, as the unpacking operation that is processed in the same place or as the item that is stored without any special type of machinery. In these cases, the simple blocks can be used.

Other useful function is to generate buffers and other spaces with specific dimensions (weight and length). Inserted block is "*dynamic block*" which has some geometric properties given as variable (see Section 4.5.2).

<u>Setup of the IDS properties:</u>

IDS.DB.RESOURCES.SPACE.<u>***width***</u> = value

IDS.DB.RESOURCES.SPACE.<u>***length***</u> = value

IDS.CAD.FACILITY.FACILITY_PLACEMENT.<u>***Generate_space_as_rectangulars***</u> = true.

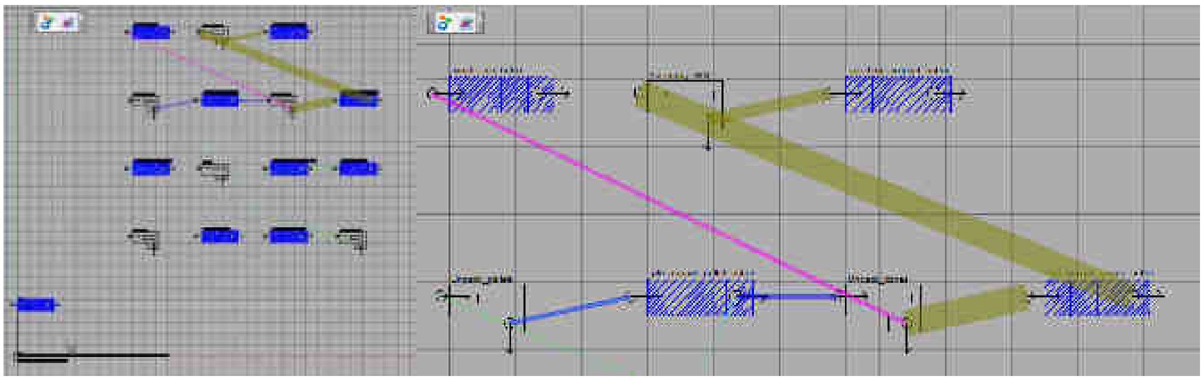


Figure 225 - Generated AutoCAD layout

In Figure 226, there is a snapshot of the layout and of the 3D. Blocks are overlapping each other, buffers (spaces) have defined sizes. **Constrained layouts** (limited) are arranged by the user into a position, as shown in Figure 227.

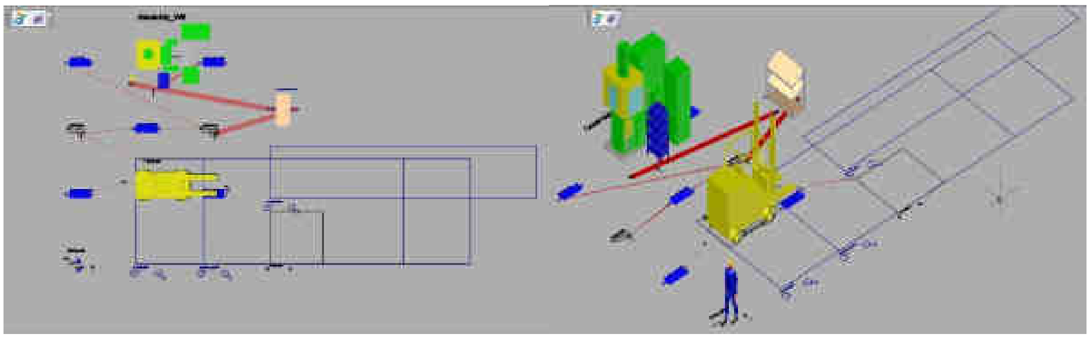


Figure 226 - AutoCAD layout, ideal with realistic shapes

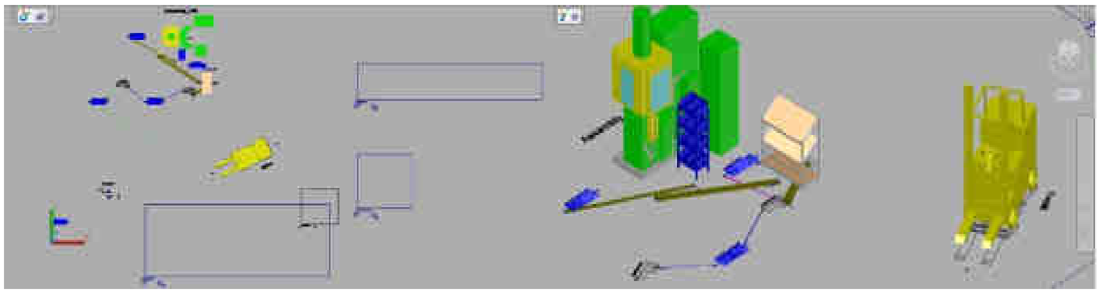


Figure 227 - AutoCAD layout, realistic layout

IDS provides several helpful tools for layout design:

- The generation of material flows
- The generation of part-in-buffers to design

A.4.12.4 Material flows

As mentioned in Section A.4.4, several kinds of material flows can be displayed: the ideal and the limited. They are either based on the kind of parts or crates, summarized flows, transportation flows or based on used units (weight, frequency, cost).

The ideal flows are automatically generated as straight line shapes, while limited flows are generated as straight polylines and the user can adjust them manually in order to obtain their final shape (curve). The example of ideal flows is in Figure 226. The example of the limited flows is in Figure 228 (left image). Vertexes of polylines are edited to achieve the appropriate shape, taking into account other resource objects or building shapes. On the right image of this figure, there is a transportation system where each colour represents a **different kind of transportation unit or crate** (truck – red, yellow; pallets – green; boxes – blue; parts – dark-yellow).

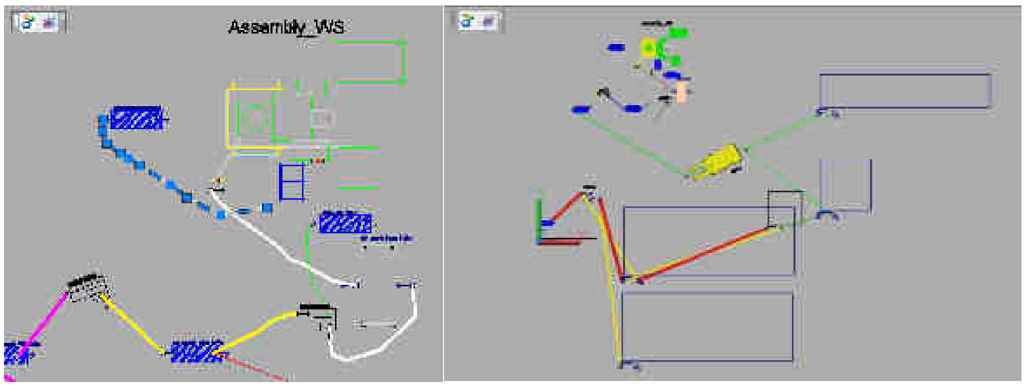


Figure 228 - AutoCAD layout, limited flows and flows based on crates

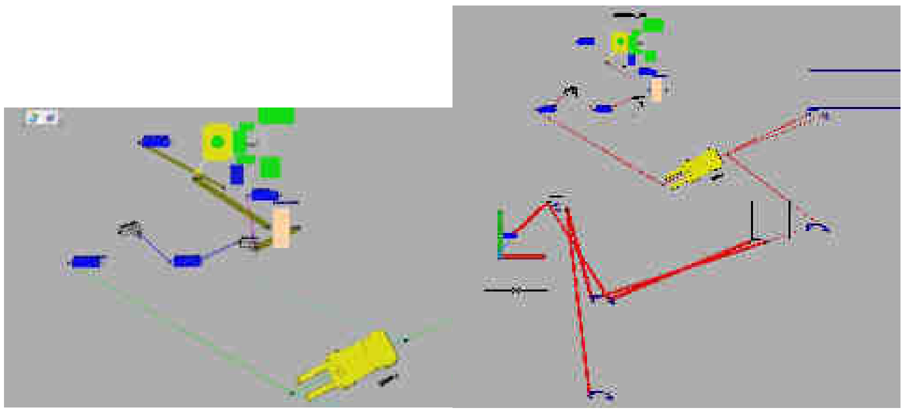


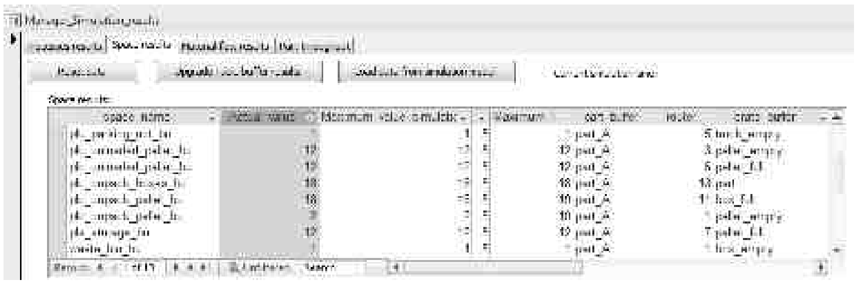
Figure 229 - AutoCAD layout, limited flows and flows based on crates

In Figure 229, flows based on the units are: frequency (left image) and weight (right image). The frequency of movement or number of movements per time unit is the highest in the workstation (dark-yellow colour) compared with a low number of movement (12 times per shift) by forklift (light green colour). The second image displays the weight flows: red colours flows for heavy trucks (full loaded, total weight) compared with the light movement of pallets, boxes and parts (see Figure 223: Table of weights).

A.4.12.5 Part-in-buffers

Buffer sizing is also a very common task in the layout design. Based on the simulation model and subsequent results analysis, it is possible to establish the required buffer size for the system (see Figure 230, form **IDS.DB.RESULTS.SPACE_RESULTS.*Maximum value simulation***). In order to check space availability, the appropriate number of part blocks can be generated in the layout, on the proper buffers (see Section 4.7.3, form **IDS.CAD.FACILITY.BUFFER_SOLUTION**). After inserting blocks representing part or crates, it is necessary to place them in the correct place (see Figure 231) where two pallets with boxes are inserted.

In the case of *„Unpacking_operation()"*, it is possible to delete loads (e.g. boxes) and let only carrier (pallets) in buffers because boxes lie on the pallets.



space_name	Actual value	Maximum value simulation		Maximum	part buffer	route	crate buffer
pl_[illegible]_in	[illegible]	1	5	[illegible]	part_A		5 truck_empty
pl_unloaded_pallet_b	12	12	5	12	part_A		3 pallet_empty
pl_unloaded_pallet_b	12	12	5	12	part_A		6 pallet_full
pl_unpack_boxes_b	18	18	5	18	part_A		18 part
pl_unpack_pallet_b	18	18	5	10	part_A		11 box_full
pl_unpack_pallet_b	3	3	5	10	part_A		1 pallet_empty
pl_storage_in	12	12	5	12	part_A		7 pallet_full
waste_line_b	1	1	5	1	part_A		1 box_empty

Figure 230 - Buffer sizing

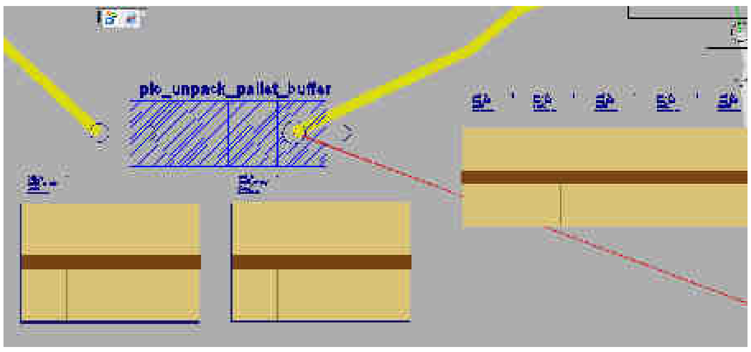


Figure 231 - Buffer sizing

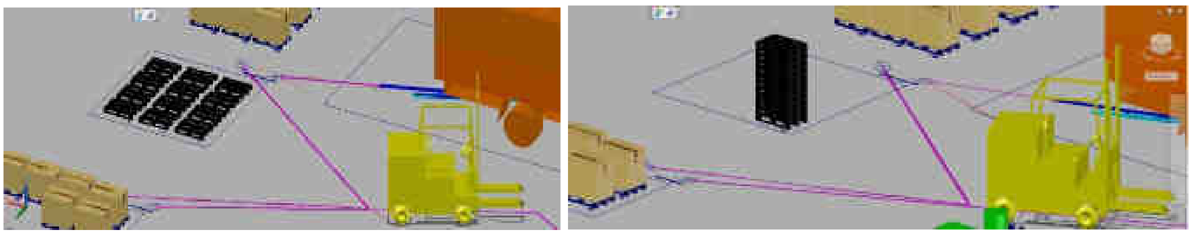


Figure 232 - Objects arrangement

For the position of objects, the user must use his knowledge and experiences. For example, it is is not convenient to store automatically inserted empty pallets side by side, but in stocks (see Figure 232) . This is an example that IDS is helpful for designing systems. But the proposed system must be checked and improved by the user. Also, it is necessary to understand the received results and data.

Appendix 5 Images

A.5.1 3D animation images (Cachapuz)

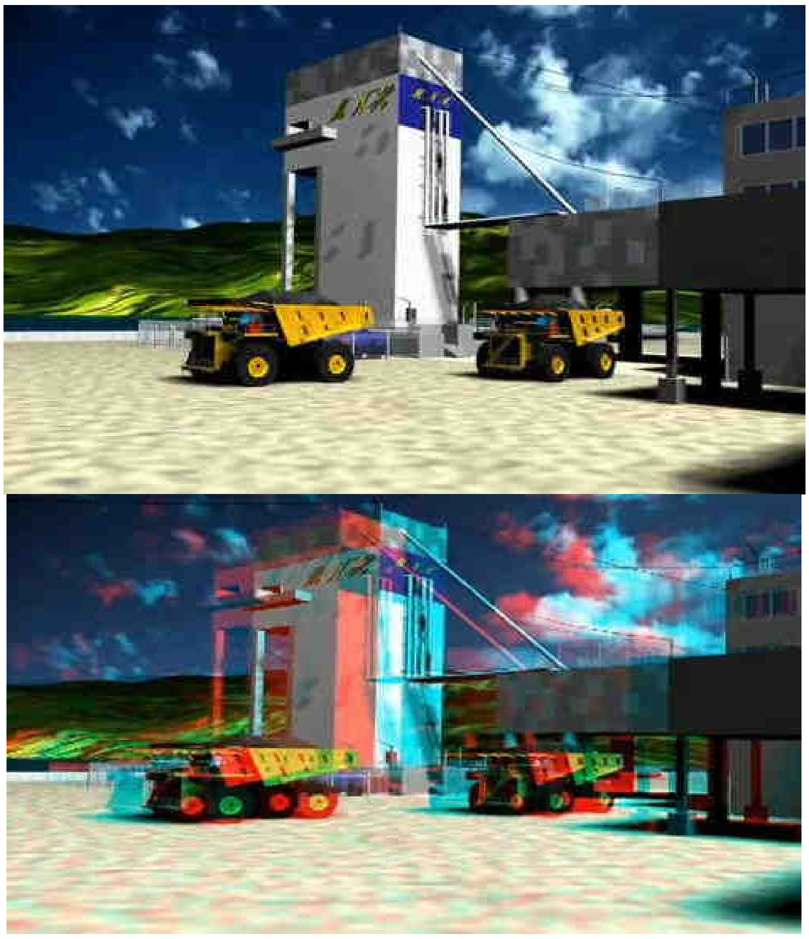


Figure 233 - Realistic Animation and Anaglyph (Stereoscopic)

Complete animations are avalaible on webpages www.youtube.com/watch?v=xrirkwckrtQ and www.youtube.com/watch?v=GLlbof30FxE.

A.5.2 Blaupunkt Bosch – Design of milk-run system

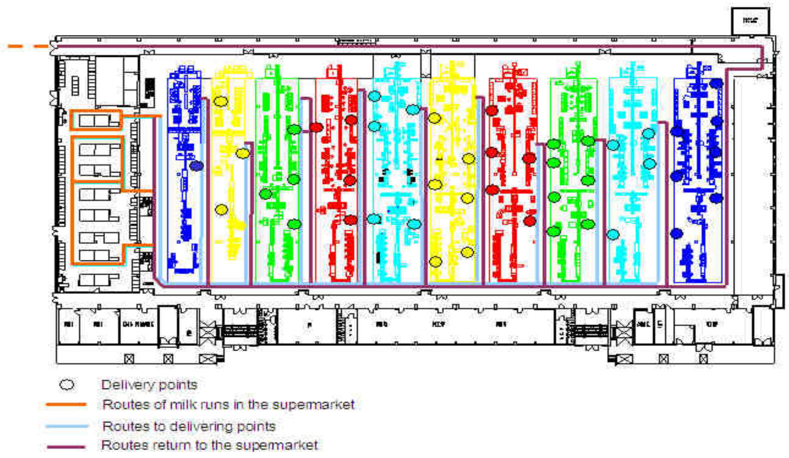

○ Delivery points
Routes of milk runs in the supermarket
Routes to delivering points
Routes return to the supermarket

Figure 234 - AutoCAD layout with routes

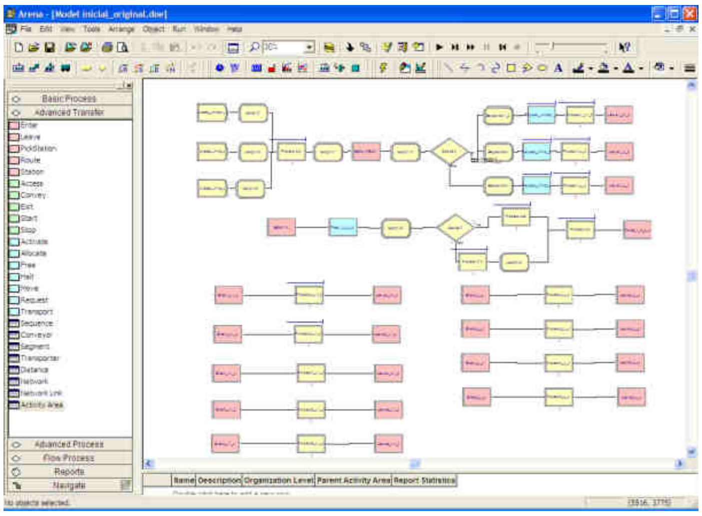


Figure 235 - Arena Simulation model

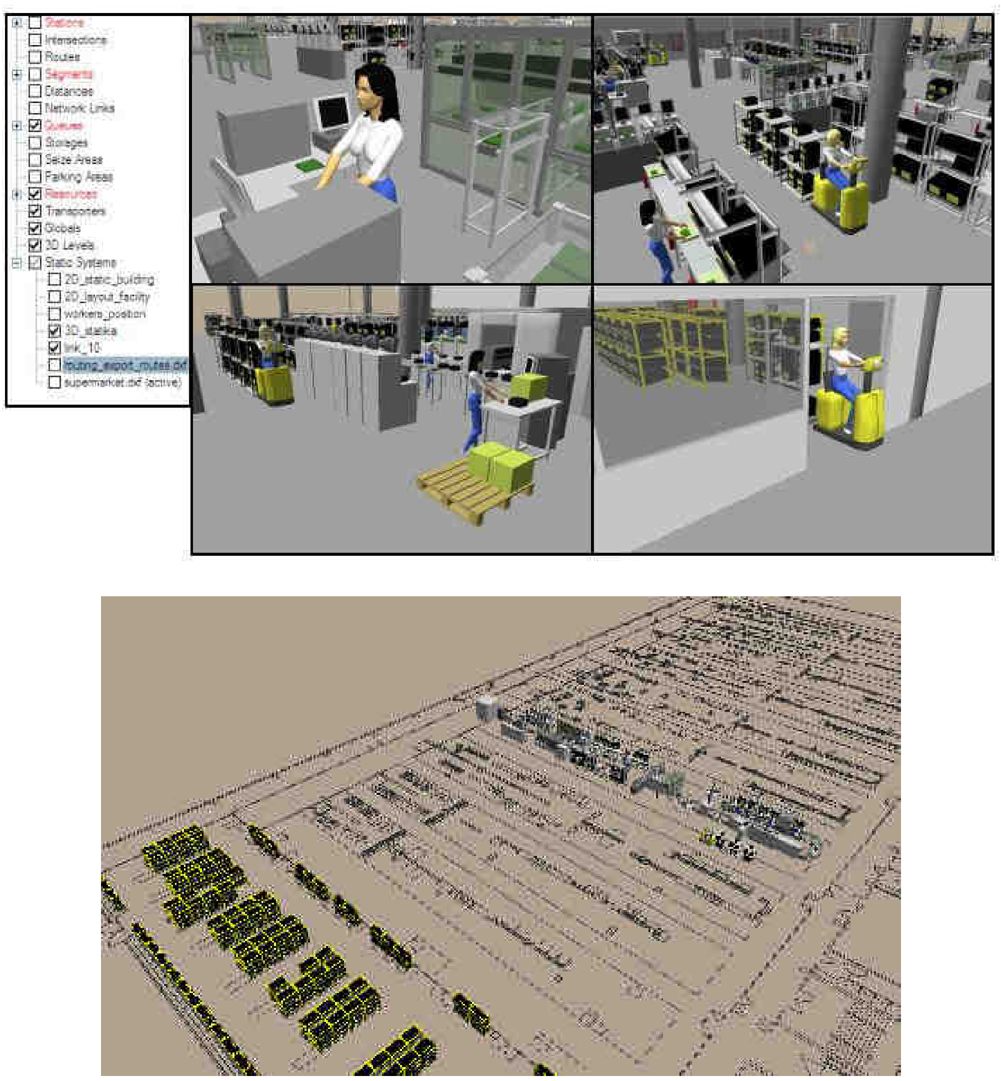


Figure 236 - Snapshots from 3D Player – realistic animations