Creating Responsive Information Systems with the Help of SSADM

Outline of a methodology to integrate performance and software engineering within a structured method

Bálint Molnár

Information Technology Foundation of Hungarian Academy of Sciences
H-1525 Budapest 114. P.O.B. 49, Hungary,
Telephone: +36 1 169-9499, Fax: +36 1 1695-395
e-mail: h4445mol@ella.hu, H4445Mol@HUELLA.BITNET,
molnar@camel.itf.kfki.hu

Errol Simon

University of Wolverhampton
School of Computing and Information Technology
Wulfrun Street, Wolverhampton WV1 1SB, United Kingdom
Telephone: +44 902-322616, Fax: +44 902-322680
e-mail: cm1951@ccub.wlv.ac.uk

Abstract. In this paper, a program for a research is outlined. Firstly, the concept of responsive information systems is defined and then the notion of the capacity planning and software performance engineering is clarified. Secondly, the purpose of the proposed methodology of capacity planning, the interface to information systems analysis and development methodologies (SSADM), the advantage of knowledge-based approach is discussed. The interfaces to CASE tools more precisely to data dictionaries or repositories (IRDS) are examined in the context of a certain systems analysis and design methodology (e.g. SSADM).

I. Introduction

The software performance engineering can be considered as "lost knowledge" in the system analyst and designer community. The software developers were involved in the early years of computing. The storage space used up and the time required to run the programs had to be carefully controlled to force the programs into the relatively small machines. As the performance of the hardware grew, the performance engineering and modelling did not get enough attention only on those fields where the strict performance requirements made it cost-effective (e.g. flight-control, mission critical embedded systems).

At the information systems engineering field, the "fix-it-later" approach proliferated; the early structured systems analysis and design methods (e.g. [Yourdon75], [Longworth86], [Brodie82], [Cameron83], [Jackson82], etc.) deferred the performance considerations to the technical and physical design or implementation stage.

The other side of the problem is that relatively few experts are available and many analysts and designers who need their services; these services can be considered especially valuable in a country with an ageing equipment base and shortage of capital to renew that base or to procure the most sophisticated and advanced equipment.

In the following sections, we outline a methodology to integrate a structured analysis and method (namely SSADM, [NCC90]) and the performance engineering methods.

II. The concept of the capacity management

The aims of the capacity management

- appropriately sizing the hardware resources in order
  - to run new applications
  - to meet the service level commitments
  - to maintain the service level of all existing systems

The capacity planning is part of capacity management and it is used to predict the capacity of resources needed to support the existing applications as they are grow and new applications as they are implemented.
Capacity planning techniques are used to influence the design of application systems to optimise the performance of applications sharing the same hardware with existing systems thus helping organisations to make the most effective and efficient use of hardware resources within the limits of the budget.
The SSADM, MERISE, SDM, Information Engineering, etc. ([NCC90], [Matheron90], [Turner90], [Martin81]) widely known structured methods concentrate on the functionality of information systems; they generally do not have a peculiar technique or procedure for assessment of the level of performance. However, these methods collect a lot of non-functional requirements and service level demands systematically and steadily during the various stages of the systems analysis and design.
So there is an opportunity to ground the performance modelling on these hard data, naturally the accuracy of the prediction would fit to the available knowledge-level; so this model would be refined the same way as the design of information system.

III. The Notion of a Design Space

A multidimensional design space classifies the software system architecture. Each dimension of a design space represents a design alternative or options of the systems, within a dimension the variations or range of the possible system characteristic is shown; the values along a dimension correspond to alternative requirements or design choices. For example, the required response time could be a dimension, the values of this concrete dimension can be continuous but it may take up discrete values (low, medium, high); in most cases the content of a dimension is a few discrete values corresponding the design alternatives and need not possess any useful metric (distance measure). For example, a dimension that represents a structural choice (physical design or initial system decomposition) is likely to have a discrete set of possible values, which may or may not have any meaningful meaning. Methods for specifying the behaviour aspect of an information system include entity-event modelling, state transition diagrams, Petri-nets, etc., they represent logical or technical design dimension where the different techniques display the variations along the dimension.
The different dimensions are not necessarily independent, in fact, it is important to discover correlation between the dimensions to create design rules describing appropriate and inappropriate combinations of choice.
A key part of the design space approach is to choose some dimension that reflects requirements or evaluation criteria (e.g. performance or functional) while other dimensions reflect structures. Then any correlation found between these dimensions can provide direct guidance: they show which design choices are most likely to meet the functional and non-functional requirements to a new system.

The requirement specifications can be considered as the functional (sub)space while the results of the technical and physical design stage during the system decomposition can be regarded as the structural design (sub)space.

IV. A design and modelling methodology of capacity planning

As we mentioned above, the idea is to integrate performance engineering into software engineering.
The main stages of the proposed methodology:

- specification of the information systems
- performance model construction and evaluation
- analysis of the model and feedback to the systems specification
- alternative specification and refinement of the design
- steady verification and validation of the performance model

There is a so called subject guide [CCTA90] that provides the skeleton of a methodology and yields some clues how to interface SSADM and performance engineering but it is not sufficiently detailed.

A. The specification

During the analysis and design stages of information systems, the analyst collects data about

- the functional and non-functional requirements (service level, security, etc.);
- the designer supplies the software architecture attributes,
- the initial software structures and the proposed hardware configuration.

SSADM provides several methods to refine the design and various models of information systems and at the selection of the Business System Options (BSO) and Technical System Options there are points where the capacity planning can offer assistance in assessing how well the new application will work. Hares stresses that "the logical design = physical design" equivalence [Hares90] in contrast of the "fix-it-later" general approach and this means that we should incorporate the performance estimating in the method, more exactly, every deliverable should be subjected to a performance modelling activity to refine the design in terms of non-functional services. The responsiveness of an information system [Smith90] means the response time or throughput as seen by the users. Nevertheless, the main aim of the performance engineering is not an ideal performance (not cared about the costs) but it wants to achieve a performance that is cost-effective and fits into the user functional and non-functional requirements.
The general experience shows that the tuning of the physical design and the implementation is too late.
To assess the application's responsiveness, the specification should provide sufficient performance data. The Logical Data Structure, the Function Catalogue, the Requirements Catalogue contains volumetric and volatility data and several cross-references.
The performance requirements must be specific and measurable ("rapid response is required" is vague statement); they appear in the form of non-functional requirements in Requirements Catalogue, during the analysis, the accuracy of the requirements' data and the hard fact, functional requirements should be steadily refined. This procedure corresponds to the selection along the dimensions of functional design space.

The structural dimensions include the hardware configuration data (processors, network, etc.) and initial software structure, module decomposition.
The software model specification consists of:

- performance goals
- workload specifications
- software execution structure
- execution environment
- resource usage

The workload specification means the determination of

- system uses or requests for system function
- the rate at which each is requested
- any special patterns of requests

This information can be gained from the entity-event model and function definition in SSADM.
The software execution structure makes up

- the software components that execute
- the order of execution
- component repetition and conditional execution

The execution environment

- hardware configuration
- abstract machine - operating system, other support software

Resource usage

- the number of instructions to be executed
- the number of I/O operations for each device
- the number and types of abstract machine service routines
- the amount of memory for code and for data

V. Model Construction and Evaluation

There is a variety of performance modelling techniques:

- queuing model [Kleinrock75]
- Petri net model [Peterson77]
- execution graph models [Smith90]

The analyst faces several choices, e.g.

- fix an appropriate performance modelling technique and tool for the whole project
- or select among the various methods and tools that seem suitable for a stage or step in the structured methods or use several one at the same time.

A methodology should provide guidance in this question and should supply a detailed explanation how to use the results of the analysis and in which model should be used up; that is, the interface points should be precisely defined.

A. SSADM interface points

The SSADM subject guide proposes [CCTA90] that the first performance engineering activity should be carried out at the so called Technical Options stage to create a workload model. But if we are serious about the equivalence principle (logical design = physical design) we should start the performance assessment at an earlier stage, namely at the Business System Options stage to create preliminary performance predictions. What is the base on which we establish our rough calculations?

There is a first cut logical data model and logicalised data flow model enhanced with the functional requirements and attached with modest volumetric data. The analyst should be committed to collect the facts about the planned workload and the service level details at this early stage in such a detail. To work out the alternatives and to justify them, we need some approximation of the system's capacity.
Summarising, the needed information to effectively satisfy the service level demands are:

- to construct (initial) workload model
- Service Level Requirements attached to the single workloads
- Technical Environment Descriptions (at least vague idea of the planned hardware and software circumstances)

The capacity planning exercise underpins the selection between the different Business System Options. Thereby the unreasonable and excessive ch oices can be avoided at the early phase in the project.
The above mentioned information is available in a much more detailed format at the later stages. Namely, the Technical System Option is the first point before the Logical Design where the analysis achieved a certain depth, more precisely, three perspectives of an information system are thoroughly investigated:

- the data oriented
- process or function oriented
- behaviour oriented

The different perspectives are cross-referenced according to the dichotomy principles ([NCC90], [Molnár91], [Molnár92]) in SSADM, that is there are well-grounded models for the information system to depict the functional requirements, but beside the models have strictly coupled information to them about the non-functional requirements including the service level demands.

B. Connections to the Technical Options, Logical Design and Physical Design

The Technical Options describe various ways of physically implementing the model of the information system. This is the first opportunity for capacity planning to provide outputs to the analyst to make decision and to select among the possible hardware and software environment. There is sufficient information to construct a preliminary workload for input to a capacity modelling tool that can be a general-purpose tool, such as spreadsheets and statistical analysis packages or database management package that provides calculation functions. A commercial software analysis tool can be used or a special-purpose program can be designed and created; any of them means real value to the analyst only in the case when they are integrated to a CASE tool used in the modelling activities of information system.
There is an important document the Technical Environment Description for each Technical Option, some first cut hardware and software configurations are put together in sufficient detail for capacity modelling purposes, but the later stages can modify them significantly, that are the Logical and Physical Design stage.
On selecting a Technical Option, the capacity planning techniques can be used to help assess the likely technical environments that support the functional requirements incorporated in the model of information systems and do not lessen the responsiveness according to the service level demands; the practicability and reasonability of the desired service level requirements are tested, moreover it supplies relevant inputs to the Organisational Impact and Cost/Benefit Analyses.

On calculating the data storage requirements, the analyst can use the detailed and precise volumetric information in the Entity Descriptions and Logical Design Volumes. On constructing the workload model, the analyst can estimate the arrival rates of user requests based upon:

- the Function Catalogue along with the Event frequency rate
- the Enquiry and Update Process Model
- the Dialogue Design including the Menu and Command structure

The number of accesses to each device, the resource usage, can be estimated from the event frequency rates utilising the Effect Correspondence and Entity Life History diagrams where the analysts can see the Entity Effects, the Operations in the (hierarchical) database management terms, and the Logical Success Units. These data make up the sound base for database transaction execution modelling ([Smith90]). So the analyst can decompose the functions into tasks, the tasks into logical success units, the logical success units are built up from database operations. So the processor occupancy can be computed from these data and compared to the service level requirements attached to the functions and tasks; in conflicting cases when the responsiveness of the system seems to be not implementable there should be followed some conflict resolution strategy and principles to guide the decisions when the different alternatives and trade-off situation are evaluated.
During the design of responsive information systems, the important part is continual verification of the performance model specifications and validation of model predictions.
In SSADM, the capacity planning techniques can be used to validate and verify the Logical System Design in sense of performance. The viability of running the Logical System Design on the hardware and software configuration depicted in the Technical Environment Description.
The V&V (Verification and Validation) effort matches the impact of the results and the Design Objectives regarding the Service Level Requirements. The analyst/designer should ensure that the Logical System design fulfils the non-functional demands not only the functional ones. In the Logical Design Stage, the capacity planning evaluates the completed logical design and the verification of the specification might mean changes in both models of information system and its performance model.

1. Physical Design Stage and Capacity Planning

In the Physical Design Stage, the capacity planning techniques can play a central role in developing the physical design. Based on the performance model, the analyst/designer should create performance predictions more exactly and tune design. The analyst should take into account the 80/20 rule (the 20% of the transactions yields the 80% of workloads) to construct representative workloads considering the relevant functions. The evaluation and validation of the results should be continually carried out in the same way like in the Logical System Design Stage.
The software design space can be used as a direct design guidance, if we have a set of design rules that orient the analyst/designer in the selection in the different structural, functional and capacity/performance dimensions.
This is an iterative process while the logical design is converted to physical design, the physical process specification, the physical data design and the process data interface provides sufficient information to create software execution graph. The Function Component Implementation Map gives the opportunity to make an overall estimation about the responsiveness of the information system. The detailed database

management transactions, batch jobs and utility programs can be used to make precise performance predictions, to create tune design plan and to refine the workload models.

VI. Conclusions and Research Directions

Hardware and software manufacturers are generating new products including network products at an alarming rate, the system analysts and designers are struggling how these new products can be used cost-effectively to build responsive information systems. This paper outlines the basic elements of a methodology to integrate the performance engineering and a structured method, namely SSADM. The methodology should give a steady feed-back to the modelling activities of the information system. Application of the design space concept allows to analyse the consequences and correlation of the design decisions and gives a good analytic tool to collect the relevant design rules and test them.
The design space approach can help in collecting the patterns of design knowledge and rules, after the knowledge acquisition phase this knowledge should be structured according to the KADS methodology ([Wielinga92]).
The future work can be to build a powerful tool that can employ some knowledge-based approach and integrate the performance modelling to CASE environment beyond the simple or complex performance calculations for which there are available various tools ([Simon91], [Simon92]).
After developing the capacity planning methodology, the techniques, the proposals their application and the required input will be defined in detail. The various models of the functional perspectives of the information systems are stored in a CASE tool or more precisely in the repository. The repository concept was the subject of standardisation efforts, and the premature endeavours have merit ([IRDS88], [IRDS88b], [ISO90], [ISO91]), the differences between the repository standards are matters of detail, not of basic notions. So we establish the design of a tool on the features of a standard repository and input data can come from the repository to execute the performance modelling calculations and it can be used as a fact base to knowledge-based inferences.
The knowledge-base of such a tool should store:

- knowledge of the problems to be solved
- the technologies and environment available for implementing the software (hardware, software)
- the systems analysis and design methodology, and the capacity planning techniques.

For implementation of such a system, several technologies can be used, e.g. object-oriented or expert database technology, expert systems technology. Here is needed much work to construct a precise specification of such a system and to clarify what technology would be suitable for such a modelling tool.

VII. Bibliography and References

[Brodie82] Brodie, M. L., Silva, E., 'Active and passive component modelling: ACM/PCM' in Olle, T. W., Sol, H. G., Verrijn-Stuart, A. A. (eds.), Information system design methodologies: A comparative view, Elsevier Science Publishers B. V. (North-Holland), (1982).

[Cameron83] Cameron, J.R., JSP and JSD: The Jackson Approach to Software Development, IEEE Computer. Soc., (1983).

[CCTA90] CCTA, SSADM Version 3 and Capacity Planning, Information Systems Engineering Division, CCTA, Norwich, (1990).

[Gane90] Gane, C., Computer Aided Software Engineering, the methodologies, the products and the future, Prentice-Hall, (1990).

[Essink86] Essink, L. J. B., 'A modelling approach to information system development', in Olle, T. W., Sol, H. G., Verrijn-Stuart, A. A. (eds.), Information system design methodologies: Improving the practice, Elsevier Science Publishers B. V. (North-Holland), (1986).

[Eva92] Eva, M., SSADM Version 4: A user's guide, McGraw-Hill, (1992).

[Hares90] Hares, J. S., SSADM for the Advanced Practitioner, John Wiley & Sons, Chichester, England, (1990).

[Hesse88] Hesse, W., Bosman, J. W., ten Damme, A. B. J., 'A four-level metamodel for application system development', in Bullinger, H.-J., et al. (eds.), EURINFO '88, Information Technology for Organizational Systems, Elsevier Science Publishers B. V. (North-Holland), pp 575-581, (1988).

[IRDS88] IRDS: Information Resource Dictionary System, American National Standard for Information Systems, X3.138-1988, (1988).

[IRDS88b] IRDS: Information Resource Dictionary System Services Interface, draft proposed American National Standard for Information Systems, (1988b).

[ISO90] ISO 10 0027: Information Resource Dictionary System - Framework, (1990).

[ISO91] ISO 10 0728: Information Resource Dictionary System - Services Interface, draft International Standard, (1991).

[Jackson82] Jackson, M., A., System Development, Englewood Cliffs, Prentice Hall, (1982).

[Kleinrock75] Kleinrock, L., Queuing Systems Volume 1: Theory, John Wiley and Sons, New York, 1975.

[Longworth86] Longworth, G., Nichols, D. SSADM Manual Vol. 1-2, NCC Blackwell, (1986).

[Martin81] Martin, J., Finkelstein, C., Information Engineering, Vols. 1. and 2., Prentice Hall, Englewood Cliffs, New Jersey, (1981).

[Matheron90] Matheron, J.P., Comprendre Merise, Outils Conceptuels et Organisationnels, Editions EYROLLES, (1990).

[Molnár91] Molnár, B., Frigó, J., 'Application of AI in Software and Information Engineering', Engineering Applications of Artificial Intelligence, Vol. 4, No. 6., pp 439-443, (1991).

[Molnár92] Molnár, B., 'A Framework for Reconciliation of the Meta-Structure of Repositories and Structured Methodologies', R. Mittermeir (ed.) Shifting Paradigms in Software Engineering, Springer-Verlag, Wien, (1992).

[NCC90] NCC (National Computing Centre), SSADM Manual Version Four, NCC Blackwell, (1990).

[Peterson77] Peterson, J., L., 'Petri Nets', ACM Computing Surveys, September, (1977).

[Simon91] Simon, E., S., MacEachern, P., 'Integrating Performance Modelling and CASE: A knowledge-based approach', Proc. UKCMG (1991).

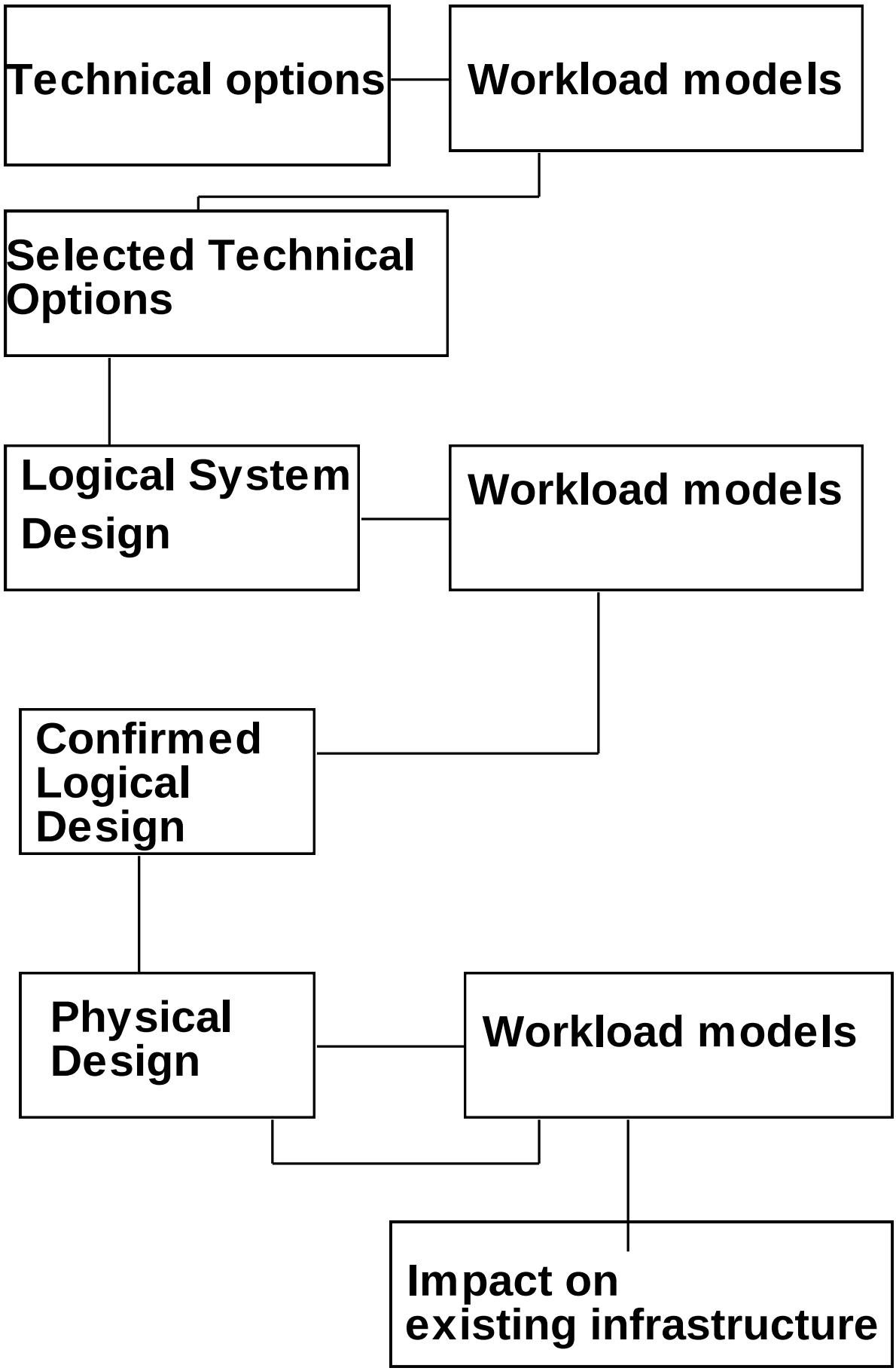
[Simon92] Simon, E., S., MacEachern, P., 'Integrating Performance Modelling and CASE', Proc. UKCMG Modelling Subgroup, pp 73-97 (1991).

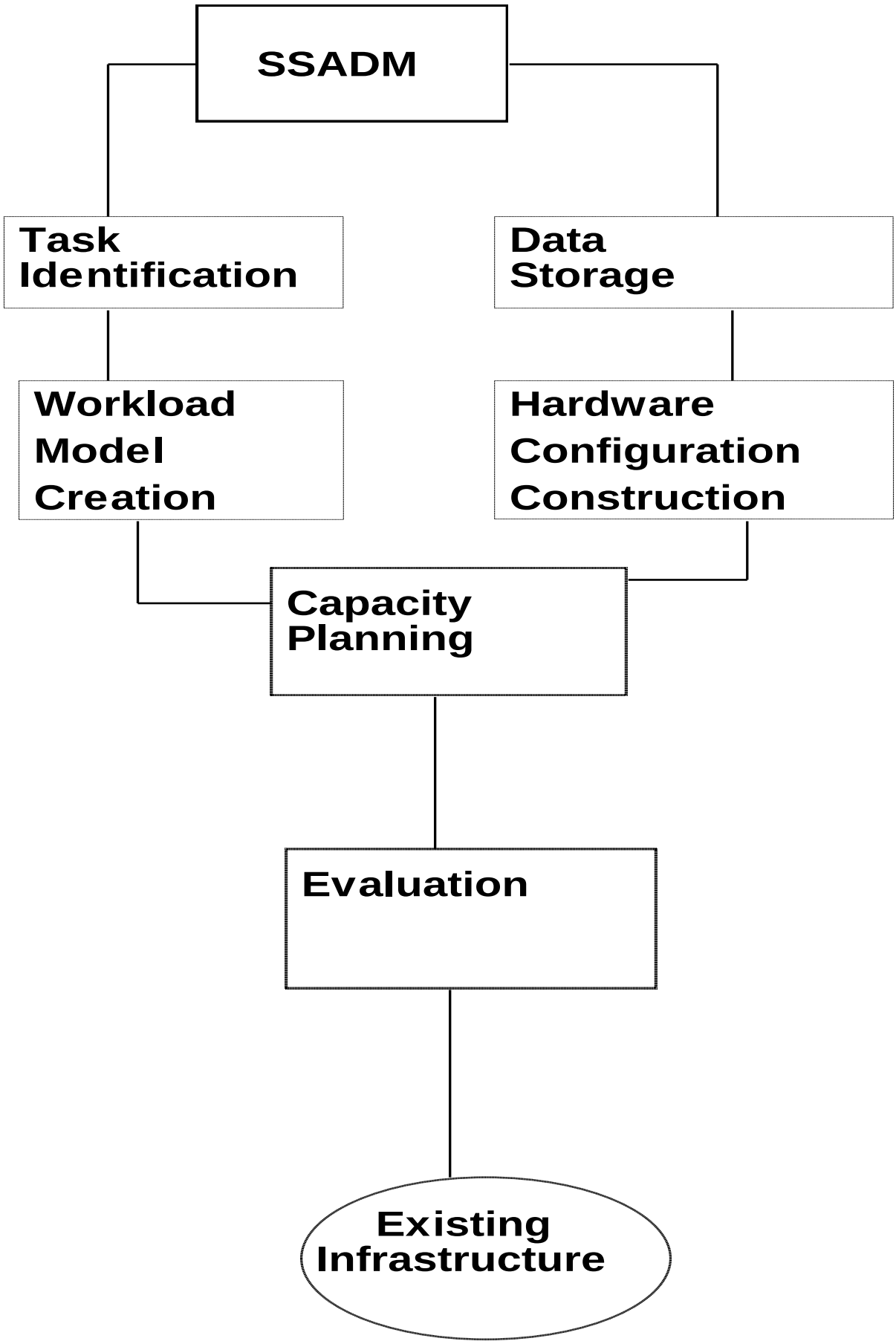
[Smith90] Smith, C., U., Performance Engineering of Software Systems, Addison-Wesley, Reading, Massachusetts, (1990).

[Turner90] Turner, W. S., Langenhorst, R. P., Hice, G. F., Eilers, H. B., Uijttenbroek, A. A., SDM system development methodology, Elsevier Science Publishers B.V. (North-Holland)/Pandata, (1990).

[Wielinga92] Wielinga, B., J., Schreiber, A. Th., Breuker, J. A., 'KADS: a modelling approach to knowledge engineering', Knowledge Acquisition, Vol. 4. No. 1, pp 5-53, (1992).

[Yourdon75] Yourdon, E., Constantine, L.L., Structured Design, Yourdon Press, (1975).


Technical options
Workload models
Selected Technical Options
Logical System Design
Workload models
Confirmed Logical Design
Physical Design
Workload models
Impact on existing infrastructure


SSADM
Task Identification
Data Storage
Workload Model Creation
Hardware Configuration Construction
Capacity Planning
Evaluation
Existing Infrastructure