Automatic Conversion of Software Specification into a Gantt-Chart subject to Organization's Constraints

Dayani M., Gelbard R.*

Information Systems Program, Bar Ilan University, Ramat-Gan 5900, Israel

Abstract

The current work introduces a model that integrates organizational constraints and policies when converting software specifications into a Gantt chart of the development process planned for that software specification. Typically, a system analyst and a software development project manager interact through software specifications, which are usually involved using a CASE tool in modeling the system. The project manager re-analyzes the model, extracts a work breakdown structure (WBS), and edits it to obtain a new Gantt chart incorporating resource allocations, cost considerations concerning outsourcing, matching each task to an appropriate qualification, setting priorities for each task, assigning safety buffers for each resource or task, setting dependencies between tasks, considering task risk assessments, identifying tasks that can be broken down into parallel subtasks, and determining organizational constraints that have to be taken into consideration.

© 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Keywords: Software project management; software specifications; software engineering; Gantt chart; software development policy; project constraints.

1. Introduction

The current paper presents a practical approach implemented as a rule-based tool that converts software specifications into a Gantt chart. Visual software modeling techniques such as a Data Flow Diagram (DFD) [1], Use Case Diagram (UCD) and others are frequently used to present the software specifications, and a Gantt chart displays the development process. This article extends previous research that introduced a method devolved to this conversion [2], [3]. However, in previous studies, the model was mostly technical, and ignored aspects such as the

* Corresponding author. Tel.: +972-3-531-8917; fax: +972-3-738-4040.
E-mail address: gelbardr@mail.biu.ac.il
development methodology habitually used by the organization, organizational policy on issues such as resource allocation and its constraints with regard to each specific development project. As a result, the Gantt chart did not represent the full process needed to develop the target software. For the Gantt chart to serve as a development plan, it must take into account all these organizational features.

This study develops a rule-based engine that represents these organizational features and applies them to the output of the technical conversion obtained from the existing conversion model. This rule engine is designed to translate organizational features such as development methodology, resource allocation policy, and specific constraints on a given development project into additional conversion rules. In turn, these conversion rules denote the order in which operations are carried out, recommends postponing or bringing forward certain operations, adding or withholding resources, and creating dependencies. Once applied to the original technical conversion, these conversion rules produce a Gantt chart for a development plan that corresponds to the specific characteristics of the organization as represented in the variables and rules defined in the rule engine.

2. Software Modeling

Software system modeling [7] is the process by which abstract models of a system are developed, where each model represents a different feature or perspective of that system. These models help analysts grasp the functionality of the system, and are used to communicate with customers. Engineers use these models to discuss design proposals and to document the system prior to implementation. In a model-driven engineering process, a complete or partial system implementation can be generated from the system model. The purpose of all analysis activity during the software life-cycle is to create a model of the system's functional requirements that is free of implementation constraints.

There are several different modeling approaches to designing systems and software applications. Nevertheless modelling is highly dependent on the specific domain to which it is applied and the nature of the problem. There are two main approaches for software modeling: the object-oriented approach and the functional approach. The functional approach involves a structured representation of the functions (activities, actions, processes, operations) that occur within the modeled system. The object-oriented approach consolidates application and database development, and transforms these into a unified data model and language environment [8]. Object-oriented modeling allows for object identification and communication while supporting data abstraction, inheritance and encapsulation. Object-oriented modeling is the process of preparing and designing the actual model code. During the construction or programming phase, modeling techniques are implemented by using a language that supports the object-oriented programming model. The main difference between object-oriented analysis and other forms of analysis is that in the object-oriented approach the requirements are organized around objects that integrate both behaviors (processes) and states (data), and are modeled after objects that exist in the real world with which the system interacts. In other traditional analysis methodologies, the process and data features are considered separately. For example, data can be modeled as ER diagrams, and behaviors can be cast as flow charts or structure charts.

Shoval et al. [6] developed an integrated methodology for the analysis and design of information systems which combines two essential software-engineering paradigms: the functional/data approach (process-oriented) and the object-oriented (OO) approach. The system analysis phase, where user requirements are set and defined, includes functional analysis and data modeling activities. This phase produces a hierarchy of data flow diagrams (DFD) and an initial OO schema which can be either created directly or from an entity relationship diagram (ERD). The design phase adheres to the OO approach, and produces a complete OO schema and a behavior schema.

2.1 COTS for Software Modeling

Here we aimed to integrate the system analysts' output into project manager software that can serve as basic work units for a Gantt chart. System analysis usually includes data/process modeling charts. One of the leading data modeling tools available on the market is Sybase PowerDesigner, which combines several standard modeling techniques [5]. PowerDesigner is a modeling and design tool that allows users to produce a range of effective solutions from defining business needs to analysis and design. It comprises a unique set of modeling tools combining several standard modeling techniques such as application modeling through UML [4], business process modeling.
and market-leading data modeling, which are all supported by a built-in meta-data management repository. One of the main features of this tool is its ability to create business process models that can be used to identify, describe and decompose business processes. This can be done on control flow or data flow at various levels of detail and focus.

3. The Prototype

The prototype was made up of three main components: (i) a User Interface to define organizational constraints. (ii) an XML parser that scans and analyzes the modeling files produced by the modeling tool. (iii) a Gantt-chart creator.

3.1 Defining Organizational Constraints

The prototype took the following functionalities into account:

- Allowing various resource combinations, including outsourcing.
- Defining risk factors, and letting the prototype's conversion tool select the activities that would be most favorable to the organization.
- Defining the minimum qualifications required for each task.
- Defining the project setup time that affects each resource first assigned to this project.
- Timing activities according to selected risk options or previous index order.
- Allowing each task to be split into parallel subtasks; in this case, the number of possible subtasks must be defined in advance for each task.
- Automatic task qualification and resource matching, in keeping with the following policies:
  - "Qualification required": Full compatibility among the skills required for the task and resource qualification.
  - "Same level of qualification or higher": Allow each task to be assigned to the same qualification resource or a higher level one.
  - "Same level of qualification or lower": Allow each task to be assigned to the same qualification resource or a lower level one.
- Various assigning methods, such as "First available" and "Qualified resource".
- Automatic outsource assigned, applied when required qualified resources are lacking.
- Setting dependencies among all project tasks to ensure that no task activity is begun before completion of the preceding one.

All these functionalities were manipulated via a specific User Interface as illustrated in Figure 1.

3.2 The XML Parser to analyse system modeling

The above-mentioned organizational constraints are executed on the tasks that are predefined within the software modeling tool. The tasks are automatically recognized by an XML parser that scans and analyzes the files produced by the modeling tool (SAP Sybase Power-Designer), in the following way:

1. Identifying the objects in all the modeling diagrams. For this prototype an XML parser was developed to parse PowerDesigner outputs.
2. Analyzing and integrating the metadata of each object identified (metadata include the description of each task; i.e. priority, risk, task type, and other details). All these parameters can be defined by the user as comments written in the comment dialogue box of each object as illustrated in Figure 2.
3. Creating a WBS (Work Breakdown Structure) for each object.
3.3 The Gantt-chart Creator

The Gantt chart is generated by MS-Project. The current module arranges all the above-mentioned parameters in an MS-Project format and executes the following automatically:

1. Generating a Gantt chart out of the WBS list, creating tasks corresponding to the analyzed properties (resource name, priorities, dependencies etc.).
2. Performing resource leveling (using the built-in project builder option) according to the priorities set previously.
3. Saving the Gantt chart in an external file.
4. Displaying the results in the user interface.

Fig. 1. User Interface to Define Organizational Constraints.
Figure 3 illustrates three sequential tasks (WBS) with no organizational constraints. By contrast Figure 4 illustrates the same three sequential tasks planned according to organizational constraints as well as according to the additional parameters defined via the dialogue boxes of these objects in the modeling tool.

Fig. 2. Adding Parameters via the Comment Dialog-Box.

Fig. 3. Three Original Sequenced Tasks

Fig. 4. Execution of Organizational Constraints and Object's Parameters
4. Discussion

This prototype has the following advantages over current tools:

1. Saving management time: The model can produce a work plan at the push of a button and will save project managers time they currently invest in this process.
2. Exploring several development alternatives: By manipulating the variables defined in the rule engine, project managers can examine different development strategies, and analyze the potential outcomes of meeting or ignoring certain constraints. The method gives project managers the opportunity to examine various scenarios (what-if analysis), and consider their implications for the resulting development plan.
3. Producing applicable development plans: The rule engine guarantees that all the features of organizational culture, as well as the capabilities and constraints are taken into consideration when producing the development plan. Because no component can be omitted or ignored, the ensuing product closely represents the envisaged development process.
4. The combination of all the above advantages should yield software development projects that closely adhere to their work plans. This means fewer deviations from originally defined schedules and resources.

References