Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 64 (2015) 73 – 78

Procedia Computer Science

Conference on ENTERprise Information Systems / International Conference on Project MANagement / Conference on Health and Social Care Information Systems and Technologies, CENTERIS / ProjMAN / HCist 2015 October 7-9, 2015

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.

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

doi:10.1016/j.procs.2015.08.465

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