The SPRINT-SMEs Approach for Software Process Improvement in Small-Medium sized Software Development Enterprises

Vassilis Gerogiannis[1] , George Kakarontzas[2], Leonidas Anthopoulos[1]
TEI of Thessaly
(1) Dept. of Business Administration
(2) Dept. of Computer Science & Engineering
Larissa, Greece
{gerogian, gkakaron, lanthopo}@teilar.gr

Stamatia Bibi, Ioannis Stamelos
Aristotle University of Thessaloniki
Department of Informatics
Thessaloniki, Greece
{sbibi,stamelos} @csd.auth.gr

***Abstract*—Software Process Improvement (SPI) of Small Medium Enterprises (SMEs) is gaining momentum in software research and industry. It is recognized that in most countries software industry is composed of a scheme made up mainly of SMEs. This paper condenses efficient Software Engineering practices that will help a SME make tangible improvements in finite time. A practical and easily applied mechanism for SPI is suggested tailored in the needs, size and type of each SME. The main concept of this process involves creating a knowledge base that consists of information about three distinct improvement areas: experience, process and product area. This knowledge base is formed as an ontology that will provide further inference knowledge on the interrelationships of the information recorded. Such knowledge is exploited to define problematic areas and test alternative improvement solutions.**

Keywords—software process improvement; Small Medium Enterprises; software engineering ;software requirements; software reuse; software estimation

I. INTRODUCTION

Software Process Improvement (SPI) of Small Medium Enterprises (SMEs) is gaining momentum in software research and industry. The SPI process is a challenging process for most SMEs aiming at preventing project failures and delivering high quality software products consistent with end-customers' needs.

The objective of SRPINT-SMEs approach (Research in Software PRocess ImprovemeNT Methodologies for Small & Medium sized Software Development Enterprises) is to propose and develop a practical framework for the improvement of software processes which take place in Small & Medium sized Software Development Enterprises (SW SMEs). The framework emphasizes process assessment and improvement practices to be adopted by software SMEs in order to increase their competitiveness. The framework concentrates on improving selected process domains such as the Requirements Engineering process (RE) and Project Planning. The following stages of a SPRINT-SME approach will be supported by the framework's analysis methods:

(i) Assessment of software process domains and selection of defective ones for further analysis and improvement

(ii) Definition of a knowledge base that better describes the software domain under improvement

(iii) Creation and further analysis of the ontology that represents the domain

(iv) Experimentation and suggestions for improvement

The framework will be particularly useful for software SMEs interested in implementing lightweight and flexible SPI projects based on critical improvement issues, tailored in cost/time/resource constraints and consistent with their individual needs.

In this paper in Section II we will point out the problems of traditional SPI methodologies when applied to SMEs and resume the SPI Initiatives for SMEs. In Section III we will present the four step analytical SPRINT- SME approach for software process improvement. In Section IV we will present a simple example of its application in improving the domain of Project Planning. Finally we conclude the paper and present ideas for future work.

II. SOFTWARE PROCESS IMPROVEMENT INITIATIVES

In general, software process improvement (SPI) approaches are either inductive (bottom-up) or prescriptive (top-down) [1]. On the one hand, inductive approaches, such as the Basili's Quality Improvement Paradigm [2], start by identifying and understanding the most critical processes in a software organization which are required to be improved. Improvement goals are, consequently, set and the process improvement is realized by a pilot project. However, inductive approaches have been criticized that they are applicable only when the organization processes are characterized by an adequate level of maturity [3]. One the other hand, prescriptive approaches, such as the Capability Maturity Model Integration (CMMI) [4] and ISO/IEC 15504 (also known as SPICE) [5], follow the "one size fits all" paradigm.

They define sets of best practices (Key Process Areas) which are required to be evaluated, regardless the characteristics or the needs of an individual software organization under assessment. A prescriptive approach supports the benchmarking of an organization's processes against these practices. A typical software process assessment/improvement project according to a prescriptive approach often demands large amount of resources and investment costs. For example, it can last between 18 to 24 months [6] and, thus, the SPI project imposes time and resource constraints which are difficult to be met by a small or a medium sized SW enterprise.

To address the above problems, in the relevant literature, the so called "lightweight" SPI approaches have been proposed [1,7]. A lightweight approach does not require extensive resources to be deployed in the SPI project and allows the flexible consideration of certain process areas which are the most critical for a specific organization (e.g. Project Management, Requirements Engineering, Testing, etc.).

However, a practical problem that remains open in most SPI approaches is that their application focuses mainly on providing answers on "what a software company should perform in order to improve its processes" and not on "how a software process improvement project will be conducted" [6]. To get answers in the second question the relevant literature proposes a number of process modeling approaches and process analysis/assessment techniques. The objective of a process modeling approach [e.g., 8, 9] is, first of all, to get an understanding of the current (as-is) process and then to propose a process redesign (to-be process) that will be more effective and efficient than the current one. The aim of analysis/assessment techniques is to perform measurements/estimations for the budget, time, scope and resource requirements of the SPI project and analyze the project results (success factors [12]) with respect to the initial project objectives [10, 11].

The SPINT-SMEs approach takes into consideration the needs of SMEs and proposes a lightweight, easily applied framework to help a SME improve domains of its process [13,14].

III. A RIGOROUS METHOD FOR SPRINT-SMES

In this section, we will introduce the SPRINT- SMEs approach for the Improvement of the Software Process of a Small Medium Company. The SPRINT- SMEs approach is a lightweight yet rigorous methodology for efficiently improving certain process domains of SMEs. In section II we have stated the specificity of SPI in SMEs and their difficulty to follow models such as CMMI or SPICE. Our approach is tailored to the needs of SMEs as it is efficient, easily adoptable, non bureaucratic and independent of company specific assets.

We suggest a four step process improvement model that consists of the following steps:

1. **Select domain for SPI**. Definition of the particular software development domain that needs improvement.
2. **Define a SPI knowledge base**. Selection of attributes that describe the particular domain and definition of metrics that better represent these attributes.
3. **Ontology analysis**. Form an ontology that describes better the relationships among the attributes of the SPI knowledge base. Application of tools and methods to better analyze an ontology that is feeded with data coming from the SME.
4. **Experimentation and improvements**. Exploit the results of the previous step to find problematic and defective domain areas. Make changes, experiment and suggest improvements.

A. *Define the Domain of Improvement*

The first step of the SPRINT-SMEs approach is to define the defective domain that will be investigated. The domain identified will then be set as the target of the improvement efforts. The starting point of popular SPI models is the specification and improvement of the quality of the total development process.

An objectively measurable specification of software process quality is a prerequisite for such types of models. However, the definition of software process quality is not always the same and does not apply to all types of companies. Additionally, the effort required to improve all aspects of software process is often prohibitive in terms of time and cost for most SMEs since they do not possess neither the know-how nor the resources to achieve such improvement goals.

Defining the software process domain that will be set under observation is a managerial decision and depends on the needs of the SME and the type of projects that it handles. For example, the domain under improvement can be decided from the traditional software lifecycle models: requirements engineering, design specification, programming and development, software testing, software development management etc.

The selection of one or more of these domains for process improvement will then define the company specific aspects that need to be specified to continue with the SPINT-SMEs approach. Some of the domains of improvement that the SPINT-SMEs approach focuses are:

- Requirements Engineering
- Software Process Estimation
- Software Project Quality
- Personnel Management

B. *Create a SPI Knowledge Base*

Target of this step is to specify and design a Knowledge Base that consists of information relevant to the knowledge required for Software Process Improvement procedures of the domain (s) pointed by the previous step. A knowledge base [17] is a database that stores data for knowledge management. Knowledge management (KM) [18, 19] comprises a range of strategies and practices used in an organisation to identify,

create, represent, distribute, and enable adoption of insights and experiences. Such insights and experiences comprise knowledge, either embodied in individuals or embedded in organisations, such as processes or practices [20].

Knowledge bases are commonly used to complement a procedure for sharing information among members of a community. They might store critical enterprise data, personnel information, process metadata, troubleshooting information, knowledge tags, or answers to frequently asked questions. Typically, a search engine is used to locate information in the system, or users may browse through a classification scheme.

Using a KM approach, knowledge created during software process can be captured, stored, disseminated and reused, so that better quality and productivity can be achieved. KM can be used to better support management activities, such as software process definition, people allocation and estimation, software development activities, such as requirement analysis and test case design, and quality assurance activities, such quality planning and control.

C. Ontology Analysis

In order to design the structure of the SPRINT-SMEs Knowledge Base we follow an ontology-based approach. In computer and information science, an ontology formally represents knowledge as a set of concepts within a domain, using a shared vocabulary to denote the types, properties and interrelationships of those concepts [21], [22].

Ontologies are structural frameworks for organizing information and they are used in artificial intelligence, Semantic Web, systems engineering, software engineering, biomedical informatics, library science, enterprise bookmarking, and information architecture as a form of knowledge representation about the "world" of interest or some part of it. The creation of domain ontologies is also fundamental to the definition and use of an enterprise architecture framework [22].

Different complementary ontologies have to be developed to address the full spectrum of knowledge in SW process improvement projects (i.e., tacit and explicit knowledge, knowledge about projects, knowledge in projects and knowledge from projects). The SPRINT-SMEs approach suggests three sub-ontologies covering three distinct process improvement knowledge domains:

- *Experience ontology*: The experience ontology describes skills and qualifications required for performing specific improvement practices.
- *Process ontology*: The process ontology enables defining a hierarchical process type structure and alternative process decompositions and dependencies (for example, it is possible to state that "Requirements Traceability" is dependent on "Requirements Specification").
- *Project content ontology*: The project ontology supports the representation of information about the improvement of the project content which includes project artifacts (e.g. source code, UML diagrams etc.) and the project content in general. Examples are: "no of class diagrams = 40", "number of use cases in the use case model = 30", "persistence framework = Hibernate" etc.

D. Experiment and Improve.

In this step we suggest the use of formal notations and tools to represent and experiment with the ontologies defined in the previous step of the approach. Some of the methods that can be used are: (i) UML for ontology representation which is a well-known standard in the software development community, (ii) clear and rigorous semantics provided by the definition of a metamodel itself, and (ii) use of tools for the generation and verification of the SPRINT-SMEs models. Especially the use of tools alleviates the difficulty of process description verbosity. For example, a human role or a phase in a process is described once and the same definition is reused, whenever the description of the same role/phase is required.

Tools and methods that can be used to further analyse and experiment with the ontology can come from the traditional group of CASE tools (such as process framework tools, simulation tools, etc.) or can be tools that provide an estimation and inference mechanism (such as probabilistic techniques, fuzzy logic based techniques, Social Network analysis tools, etc.). The SPRINT- SMEs approach, for example, focuses, among others, on two representation tools a) the open source Eclipse Process Framework (EPF) Composer and b) the Bayesian Networks Analysis tools.

In particular, a tool that provides the above mentioned capabilities is the open source Eclipse Process Framework (EPF) Composer. EPF Composer includes the following advantageous features to be exploited by the SPRINT-SMEs approach: (i) A Method Content Authoring Feature, to define process roles, tasks, work products and guidances, independently from the process definition, a feature which makes these method ingredients reusable throughout the process description. (ii) A Process Authoring feature, to describe processes as sequences of tasks (performed by roles) which produce work products. (iii) A Process Configuration Feature, for incorporating the process descriptions into packages (plugins) to be re-utilized in the definition of other processes. This functionality allows the customisation of processes to particular contexts and it very important since a SW development company may follow similar but slightly different methods and processes.

On the other hand, SPRINT-SMes exploits the advantages from powerful estimation techniques such as the Bayesian Belief Networks (BBNs) [15]. BBNs can be helpful since they can provide: i) a way to represent project/process attributes and identify their interrelationships, ii) capabilities for multiple attribute estimations, iii) results indicating confidence of the estimation, iv) solutions that can be easily interpreted and confirmed by intuition, and v) a formal method that can be used alone or combined with expert judgment. A methodology that can be used with the support of Bayesian Network Analysis Tools consists of three phases, namely [41]: i) metrics collection of the current ontology, ii) selection, application and evaluation models expressed in BBN terms, and ii) specification of a new improved process.

IV. AN ILLUSTRATIVE EXAMPLE

In this section we will present a simple application of the SPRINT- SME approach. We apply the four step methodology in order to verify the applicability of the proposed approach.

The *first step* of the approach involves the identification of the domain under improvement. We isolated project planning phase as the target of the improvement attempts. We selected to use this domain in our example as it is pointed out to be a very defective task that cannot be easily performed in SMEs. During project planning the project process need to be defined along with the schedule and its activities. People to perform the project activities have to be allocated. Also project monitoring and control should be performed. This involves tracking the accomplishment of the activities and managing the necessary time to perform them. Software project planning involves activities such as:

- Project process selection: This might involve the selection of standard processes such as RUP, SCRUM, ICONIX or even hybrid methods that fit the particular needs of the company.
- Resource allocation: This task involves the selection of the development team, the allocation of people to tasks. Also in this task the selection of the necessary software tools and hardware equipments is performed.
- Project monitoring and controlling: Usually involves the necessary estimations relevant to the effort or productivity required to complete the project.

The *next step* in the SPRINT-SMEs approach is to define a knowledge base relevant to the domain under improvement. In order to create such a knowledge base, an SME is advised to use each own empirical data coming from historical projects. In case such data are not available we suggest the use of publicly available data such as those coming from ISBSG[1].

In the generic example we use metrics and data coming from ISBSG database. It is highly possible that a company that wants to estimate several aspects of software development will not possess a sufficient quantity of its own data. Therefore, using cross company data is a starting point in order to manage and estimate a software development process. Additionally, cross company data can be useful when a company is adopting a new technology and lacks experience. Cross company data may contain projects utilizing the particular technology and can provide support on estimation and implementation issues. Table 1 summarizes some of the metrics that can be used to form the project planning knowledge base.

The *third step* of the SPRINT- SMEs approach involves the identification of an ontology relevant to the project planning phase. The Software Process Ontology (SPO) originally developed in [23] was built aiming at establishing a common conceptualization for software organizations to "talk" about software processes. It was divided into four sub-ontologies, namely: activity, resource, procedure and software process ontologies. Figure 1 shows a fragment of the first version of this ontology that includes concepts from the activity, resource, and software process sub-ontologies. We chose this fragment, because we are interested in the portion of the SPO conceptualization that is more relevant for project planning.

TABLE I. METRICS IN THE PROJECT PLANNING KNOWLEDGE BASE

Metric	**Possible Values**
Function Points	Continuous values
Max team size	Continuous values
Development Type	NewDevelopment ,Re-development, Enhancement
Language Type	3GL, 4GL, ApG
Primary Programming Language	ApG, 4GL, ACCESS, C, C++, CLIPPER, COBOL, CSP, EASYTRIEVE, JAVA, NATURAL, ORACLE OTHER, PERIPHONICS, PL/I, POWERBUILDER, SQL, TELON, VISUAL BASIC
Organisation Type	Aerospace/Automotive, Banking, Communication, Community.Services, Computers, Electricity,Gas,Water, Financial.Business, Government, Insurance, Manufacturing, OTHER, Professional Services, PublicAdministration, Transport&Storage, Wholesale&Retail Trade, Defence, Electronics.
Database Management System	ACCESS, ADABAS, DB2 ,IMS, OBJECTSTOR, ORACLE, OTHER
Development Platform	MF, MR, PC
How Methodolgy Acquired	Developed/purchased, Developed Inhouse, Purchased
Application Type	DSS, Elect.Data.Interch., Executive.I.S, MIS, Network.M, Office.I.S, OTHER, Process.Control, Real-time, Transaction/Production
Business Area Type	Accounting, Banking, Engineering, Financial, FineEnforcement, Insurance, Inventory, Legal, Logistics, Manufacturing, OTHER, Personnel, Research&Development, Sales&Marketing, Telecommunications
Implementation Date	1989- 2001

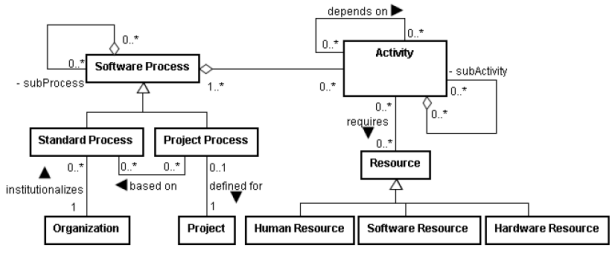

Fig. 1. Software process ontology [23].

The ontology of figure 1 describes the procedure to define a software process for a project. The project manager should identify the activities that have to be performed in order to

[1] http://www.isbsg.org/

achieve the project goals. This is done by tailoring organizational standard processes, taking the project particularities and team features into account. The project process is the basis for the further project management activities. After defining the process, the project manager has to create a network of activities, define how long each activity will last, and allocate people to perform it. For a good understanding of these tasks, we need a shared conceptualization regarding software processes.

Based on the ontology presented in figure 1 we use Bayesian Networks in order to experiment with the data represented in the ontology and find useful relationships among them to gain insights about how the project planning process can be improved. For this reason, we replace each class defined in the UML diagram of figure 1 by the appropriate metrics of table 1 or use complementary metrics if needed.

The node software process can be accumulated by software metrics relevant to the effort required to complete a software project. These can be the effort, the software size, the lines of code. The node Standard Process can be represented with the metrics that show conformance to RUP, ICONIX or XP process models while the node Project Process can represent the use of a customized variation of these standard processes for a specific project. The node Organization can be represented by metrics describing the SME. Such metrics may include the size of the organization, the years of experience, the organization type. The node Project can be identified with project specific metrics such as development type, business area type, Web development. The Activity node can be represented with the standard activities performed in software development like Planning, Specification, Design, Build, Implementation and Testing. Depending on what aspect of project planning have to be improved, these nodes can represent the relevant quality metrics for each activity or effort metrics for each activity. The node Human Resources can be represenetd with metrics like the team size, while the node Software Resources can be represented by metrics such as Use of Case Tools, Programming Language, Data Base. Finally, the node Hardware can be replaced by metrics such as the Development Platform and the Architecture type.

Figure 2 represents the Bayesian Network as it is formed after the representations mentioned previously. This Bayes Network can then be useful for applying inference. For example, specific tools can be used to redefine the structure of the network based on data from real projects. These data can be further analyzed to create probability tables that show how each node can affect the neighbor ones. Certain inferences can be made to show how the change in the values of a metric can affect the values of another metric and, finally, reach some conclusions regarding good and bad practices in software project planning.

V. CONCLUSIONS

This paper has presented in brief a rigorous approach to systematically model and guide Software Process Improvement for SMEs. The SPRINT- SMEs approach is based on a four step assessment and improvement process of a particular software process domain. The proposed methodology takes into consideration the characteristics and the needs of the individual software organization under assessment and does not demand a large amount of resources and investment costs.

As future work the proposed framework will be validated at a multiple case study involving dynamic Greek SW SMEs, which show a constant interest in redesigning and improving their development practices. Thus, the SPRINT-SMEs project will result in a set of best practices that will constitute a publicly available guide for other SW SMEs interested in gaining competitive advantages by changing their role from bespoke to market-driven software product developers.

ACKNOWLEDGMENT

This research has been co-financed by the European Union (European Social Fund – ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: ARCHIMEDES III. Investing in knowledge society through the European Social Fund.

REFERENCES

[1] Pettersson, F., Ivarsson, M., Gorsheck, T., Ohman, P. (2008), A Practitioner's Guide to Lightweight Software Process Assessment and Improvement Planning. Journal of Systems and Software, 21(6): 972-995.

[2] Basili, V.R. (1985), Quantitative Evaluation of Software Methodology, University of Maryland, College Park, Maryland.

[3] Cattaneo, F., Fuggeta, A., Sciuto, D. (2001), Putting Coherence in Software Process Assessment and Improvement. Software Process Improvement and Practice, 6 (1): 3-22.

[4] CMMI (2002), Capability Maturity Model Integration (CMMI). CMMI for Systems Engineering, Software Engineering, Integrated Product-Process Development and Supplier Sourcing (http://www.sei.cmu.edu/cmm/cmm.html).

[5] El Emam, K., Drouin, J. N., Melo, W. (1998), SPICE: The Theory and Practice of Software Process Improvement and Capability Determination. IEEE Press.

[6] Zahran, S. (1998), Software Process Improvement: Practical Guidelines for Business Success. Addison-Wesley.

[7] Mishra,A., Mishra, D. (2006), Software Quality Assurance Models in Small and Medium Organisations: a Comparison. International Journal of Information Technology and Management, 5(1): 4-20.

[8] Software Process Engineering Metamodel (2005), www.omg.org.

[9] Kruchten. P. (2003) The Rational Unified Process: An Introduction (3nd Edition), Addison-Wesley.

[10] Wiegers, K.E., Sturzenberger, D.C. (2000), A Modular Software Process Mini-Assessment Method. IEEE Software 17(1):62-70.

[11] Richardson, I. (2002), SPI Models: What Characteristics are Required for Small Software Development Companies? Software Quality Journal, 10(2): 1573-1367.

[12] Niazi, M., Wilson, D. , Zowghi, D. (2006), Critical Success Factors for Software Process Improvement Implementation: An Empirical Study. Software Process Improvement and Practice, 11(2):193–211.

[13] Gerogiannis, V. C. & Ipsilandis, P. G. (2007), Multi Objective Analysis for Timeboxing Models of Software Development. Proceedings of the 2nd ICSOFT Conference (International Conference on Software and Data Technologies), Barcelona, Spain, pp. 145-153.

[14] Gerogiannis, V. C., Kakarontzas, G., Stamelos, I. (2006), A Unified Approach for Software Process Representation and Analysis.

Proceedings of the 1st ICSOFT Conference (International Conference on Software and Data Technologies), Setubal, Portugal, pp.127-132.

[15] Settas, D., ,Bibi, S., Sfetsos, P., Stamelos, I., Gerogiannis, V.,C. (2006), Using Bayesian Belief Networks to Model Software Project Management Antipatterns. Proceedings of the Fourth International Conference on Software Engineering, Research, Management and Applications (SERA 2006), Seattle, Washington, USA, IEEE Press, pp. 117-124.

[16] Wieringa, R., Ebert, C. (2004), Guest Editors' Introduction: RE'03: Practical Requirements Engineering Solutions. IEEE Software 21(2):16-18.

[17] Guillermo Simari, Iyad Rahwan, Argumentation in Artificial Intelligence, Springer, 2009.

[18] http://en.wikipedia.org/wiki/Knowledge_base

[19] Nonaka, Ikujiro (1991). "The knowledge creating company". Harvard Business Review 69 (6): 96–104.

[20] Nonaka, Ikujiro; von Krogh, Georg (2009). "Tacit Knowledge and Knowledge Conversion: Controversy and Advancement in Organizational Knowledge Creation Theory". Organization Science 20 (3): 635–652. doi:10.1287/orsc.1080.0412

[21] Thomas R. (June 1993). "A translation approach to portable ontology specifications". Knowledge Acquisition 5 (2): 199–220.

[22] Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C., Giannopoulou, E.: Ontology Visualization Methods - A Survey. ACM Computing Surveys, 39, 4, Article 10 (2007).

[23] Bringuente, A. C. O., Falbo, R. A., Guizzardi, G. (2011), "Using a Foundational Ontology for Reengineering a Software Process Ontology". Journal of Information and Data Management, vol. 2, n. 3, pp. 511-526.

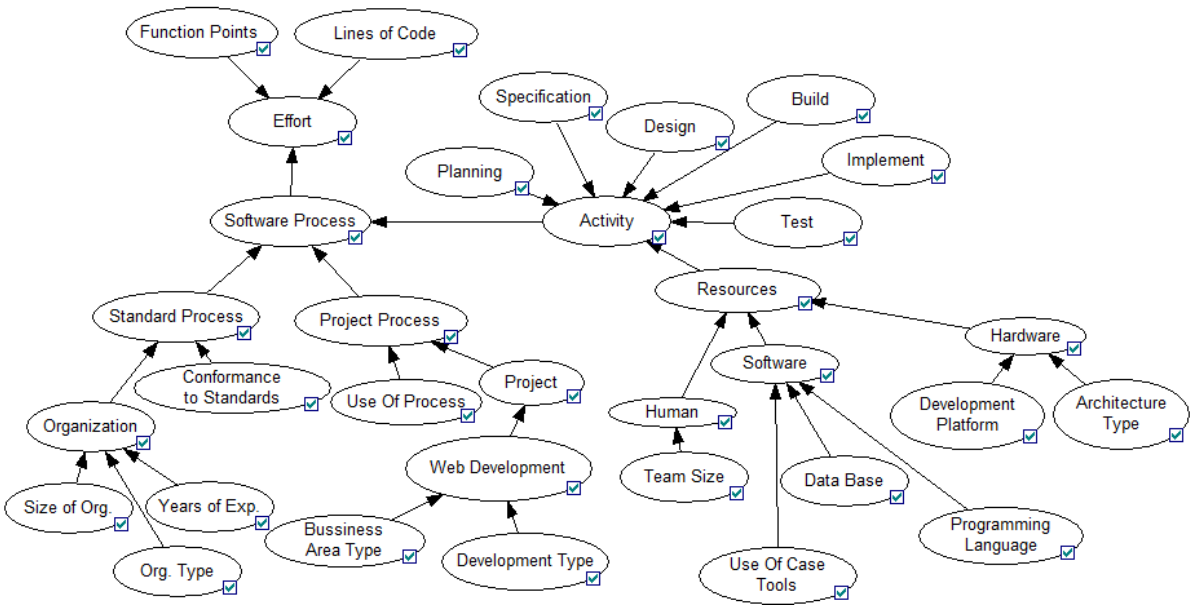

Fig. 2. A Baysian Network for the Software process Ontology presented in Figure 1.