支持过程度量的软件过程建模方法的研究*

宿为民 朱三元

(上海计算机软件技术开发中心 上海 200050)

摘要 文章提出应从过程建模的角度考虑对过程度量的支持问题,并提出一种支持过程度量的软件过程建模方法的框架.它包括面向目标的过程度量模型建模方法、支持过程度量的软件过程描述机制的选择和支持过程度量的软件过程建模算法.

关键词 过程度量,软件过程建模.

中图法分类号 TP311

软件工程概念框架由软件过程工程和软件产品工程两大分支组成[1].其中,软件过程工程的目标是生产高质量的软件过程,因而过程建模技术和过程改进技术就成为软件过程工程的研究热点.通常,过程的改进是以如下方式进行的:(1) 建立过程模型;(2) 将过程模型实例化为具体软件过程;(3) 在过程运作时发现问题;(4) 提出过程改进报告;(5) 实施过程改进.其中,第(3)步是过程改进的关键.一个有效的方法就是过程评估.目前,美国软件工程研究所(Software Engineering Institute,简称 SEI)提出的一套以其能力成熟度模型(capbility maturity model,简称 CMM)为基础的软件过程评估方法[2]已日益被广泛接受.其特有的5个能力级别成为过程不断改进的渐进式目标.1993年6月,国际标准化组织成立 SPICE 项目组,旨在开发一套软件过程评估的标准(Software Process Assessment,简称 SPA).目前,该标准的工作草案已进入第2轮投票.从 SPICE 项目组颁布的技术报告[3]来看,未来的 SPA 无疑仍是一个以 CMM 为基础的"SEI 式"的评估标准[4].

然而,这种以 SEI 的 SPA 为代表的过程评估方法却不可避免地存在若干问题:评估过程过于抽象;不同的人会因其具体评估方式不同而产生不同的评估结果;一些重要的度量数据没能及时收集等等.究其原因,就是因为没有一套定义好的、一致的评估步骤,这是由于评估与建模相脱离而造成的.如果将评估过程中所涉及的度量步骤"融化"到被评估过程的过程描述中,使需要实施的度量活动成为软件过程的一部分,必将规范度量的步骤与方法,从而解决过程 SEI 式的 SPA 存在的问题.本文正是就这一问题而展开讨论.

有关将度量与过程模型相结合的文献始见于90年代初.Pfleeger 和 McGowan 提出使用一种 SADT(structured analysis and design technique)方法描述软件过程以便发现过程中的问题,并就过程成熟度度量提供可行性建议[5,6].Lott 和 Rombach 提出一种方法将收集测量数据的活动集成到软件过程模型中[7].接着,McGowan 等人将他们的基于 SADT 的方法上升为一种基于模型的软件评估(model process assessments,简称 MBPA)方法[8],该方法描述过程建模与过程评估的结合,并将这种方法以一个"过程模型"来描述,使得 MBPA 方法进一步系统化,但仍没有解决过程评估过于抽象的问题.从1993年开始,GQM(Goal_Question_Metric)模型开始得到重视.GQM 模型是在80年代中期,由美国马里兰大学的 Victor Basisli 教授提出的一种面向目标(goal-oriented)的关于软件产品和过程测量的方法[9],它使得测量由"被动引发到过程中"变成"主动地渗透到过程中",它的由目标细化到度量的逐步求精的方法具备很强的灵活性和可操作性,因而得到了广泛的认可.相关的工作有 DeBunje 和 Saunders 通过将 GQM 度量树与一种 Enty-Task-Exit 过程树相结合,保证两树在相应节点的一致性,从而明确度量过程和步骤[10].Brochers 等人强调在将面向目标的度量引入一个软件机构时过程建模所起的

---

* 本文研究得到国家"九五"科技攻关项目基金资助.
作者宿为民,女,1967年生,博士,助理研究员,主要研究领域为软件工程,软件质量保证.朱三元,1936年生,研究员,博士生导师,主要研究领域为软件工程,软件质量保证.

本文通讯联系人:宿为民,上海 200050,上海计算机软件技术开发中心

本文 1998-03-18 收到原稿,1998-10-20 收到修改稿

重要作用[11],他们所描述的一个在软件过程模型基础上创建GQM计划的方法和步骤将对过程的度量大大地渗透到软件过程中,从而进一步增加了过程度量的可操作性.

基于以上学者的研究成果,本文首次提出应从过程建模的角度考虑对度量的支持问题,并进而提出一种支持过程度量的软件过程建模方法的框架,其基本原则是:将关于过程度量的一些特定的活动完全融入到过程模型之中,使之成为软件过程各活动(也称作"过程步")中的一种,由于所有活动是关于时间偏序的,因而随着软件过程的运作,所有度量活动将得到执行,从而保证过程度量的系统性和一致性.

下面,我们将就过程度量模型建模方法、软件过程描述机制的选择、过程建模方法等问题展开讨论.

1 过程度量模型建模方法GQM-D

1.1 GQM方法

有关研究表明,GQM方法是研究过程度量问题时的首选方法.

首先,提出度量目标(Goal,简称G),然后将该目标细化为关于过程或产品的特定问题(Question,简称Q),这些问题以度量(Metric,简称M)的方式得到回答.与McCall等人提出的FCM(Factor_Criterion_Metric)方法相比,GQM不仅支持软件产品质量的度量,而且由于GQM在将G细化为Q,M时为度量人员提供了深入到软件过程中各个细节的空间,因而可用于进行过程度量.

1.2 GQM-D方法

尽管GQM指出了产生度量的过程,但对用户而言还是过于抽象,还是不能提供发现过程中问题的依据.如果能将度量的特点与软件过程的特点相结合,将GQM进一步细化,则对用户更具指导意义.

通过考察度量的特点发现,所有的度量都有其度量的对象,即实体或实体的属性.对一个过程而言,如果仅取得过程终止时该实体或属性的度量值,则对于考察过程内部的状态没有多大帮助,更无从发现过程中间的问题,进而提供过程改进的依据.例如,某一目标经GQM细化后得到一个度量Mp:软件开发过程中开发人员的工作效率.通常的方法是利用公式

$$Mp=\frac{\sum_{n}T_{W_i}}{\sum_{n}T_{W_i}+\sum_{n}T_{B_i}}.$$

其中T_{W_i}为第i个开发人员的工作时间;T_{B_i}为第i个开发人员阻塞等待时间;n为开发人员数目.先统计所有开发人员的工作时间和等待时间,然后计算出Mp的值.

这样得到的Mp值存在两大问题:(1)由于所有的统计值是在过程结束之后得到的,经过长时间的开发,任何一个软件开发人员都很难估算出其T_W和T_B,因而计算出的Mp值的准确性值得怀疑;(2).Mp值是最终结果,当Mp值不能满足期望值时,无从考察究竟是过程中的哪一步影响了Mp值,进而发现问题.如果能将Mp的度量活动深入到开发过程中,考察每个开发人员在每个活动步后的T_W和T_B,则会得到过程的中间信息,哪一位开发人员在哪一步受到严重阻塞将会一目了然,从而发现影响效率的因素,提出过程改进的意见.深入到过程步后,计算公式将作如下改变.

$$Mp'=\frac{\sum_{m}\sum_{n}T_{W_{ij}}}{\sum_{m}\sum_{n}T_{W_{ij}}+\sum_{m}\sum_{n}T_{B_{ij}}}.$$

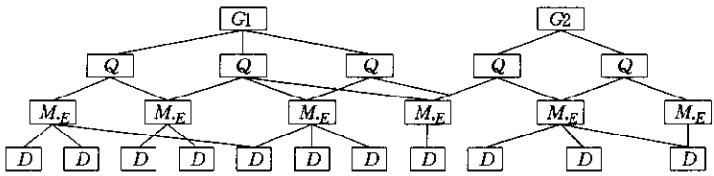
$T_{W_{ij}}$为第i个开发人员在第j个过程步的工作时间;$T_{B_{ij}}$为第i个开发人员在第j个过程步的阻塞等待时间;m为过程步数目;n为开发人员数目.同时产生了新的中间值,即

$$M_{P_{ij}}=T_{W_{ij}}/(T_{W_{ij}}+T_{B_{ij}}),$$

为第i个开发人员在第j个过程步的开发效率.

根据以上分析,我们提出GQM-D过程度量框架,它是GQM的进一步细化,如图1所示.

GQM-D对GQM的扩展主要包括以下3个方面.



其中G层为度量目标;Q层为针对G细化得到的问题;$M_{,E}$层为P经量化表示后在过程实体上的进一步分解;D层为数据层

图1　GQM/SSC度量框架

1.2.1　对度量的分解*

在以上例子中关于 Mp' 的计算实际上是 Mp 关于过程步的分解.由考察整个过程的效率分解为考察每个过程步的效率,从而发现开发人员在过程中间存在的工作效率问题.

由于软件过程有多种类型的实体,因而对一个软件过程所进行的度量 M 有可能在多种类型的实体上得到分解,这包括 M 的过程步分解($M._A$)、产品分解($M._P$)、资源分解($M._R$)和角色分解($M._J$)等等.在实施具体的分解时,可能会依据具体情况对具体的实体实例进行分解,如当实施 M 的产品分解时,可能只需针对文档类产品进行度量,记作 $M._{P.D}$,这需要在关于度量 $M._{P.D}$ 的文档中给出描述.

软件过程的一个度量 M,可能同时对多种实体类型实施分解,例如,设该过程的实体类型为 E_1,E_2,E_3,则

(1) 可进行 M 的 E_1,E_2 顺序分解,表示先进行 M 的 E_1 分解,再进行 M 的 E_2 分解,记作 $M._{(E_1/E_2)}$;

(2) 可进行 M 的 E_1,E_2 并列分解,表示 M 同时在 E_1 和 E_2 上实施分解,记作 $M._{(E_1+E_2)}$;

(3) 可进行 M 的综合分解,记作 $M._{(E_1+E_2/E_3)}$,表示 M 先同时在 E_1,E_2 上实施分解,对 E_2 分解后再对 E_3 实施分解.

在关于 Mp 的例子中,我们实际上是对 Mp 作了关于实体类型"开发人员($P.D$)"和"过程步"的顺序分解,记作 $Mp._{(P.D/A)}$.

1.2.2　引入数据项分层 D

在进行度量 M 的各种分解后,其计算方式、所使用的数据将产生新的变化.数据项分层 D 就是用于向有关的度量提供计算用的测量数据层.

在关于 Mp 的例子中,$T_{W_{ij}},T_{B_{ij}}$ 是用于向度量 $Mp._{(P.D/A)}$ 提供计算用的数据,因而放在数据项层,如图2所示.

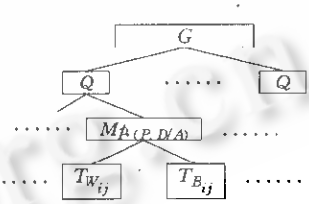


图2　GQM/SSC度量框架中Mp的分解

有些度量,其自身可能就是直接的测量数据,如度量"IINL:代码嵌套层数",对于这种情况,我们仍引入数据层,这时,D 与 M 是一致的.

1.2.3　增加度量集和数据项集的结构化描述

在实施一个度量时,度量定义的合理与否是一个关键问题,而对度量定义及其测量数据是否有一致的理解则是另一个关键问题,只有在理解一致的基础上,才能保证所收集数据的准确性和一致性.对度量和数据项进行结构化描述的目的就是帮助进行度量和定义数据,保证理解的一致性.下面定义用于规范度量和数据项描述的结构.

定义 1(GQM-D 度量集). GQM-D 度量集(记作 M-Set)中的每个度量都是一个八元组.

$$M_i=(N,C,Q,T,D,F,V,M'),$$

其中 N 为度量名,具有唯一性;C 为度量的成本;Q 为与本度量相关的问题 Q_i;T 为与本度量相关的工具,如数据存储、收集工具,分析工具等;D 为计算本度量所需的数据项;F 为本度量的计算公式或步骤;V 为本度量的期望值;M' 为本度量的实体类型分解式(如果有).

---

* 在实际操作中,也存在对 Q 的分解,产生其子问题.由于这些子问题的变化可由其度量表示,因此可将这些子问题"上升"为若干问题,与其他问题并列,只研究其度量即可.

定义 2(FQM/SSC 数据项集). GQM-D 数据项集(记作 D-Set)中的每个数据项都是一个八元组.

$$D_i=(N,M,De,T,C,P,S,V),$$

其中 N 为数据项名,具有唯一性;M 为相关的度量;De 为本数据项的定义;T 为数据收集时间;C 为数据收集人;P 为数据收集方法;S 为数据存储点;V 为数据类型. 有以下两点需作说明.

(1) 由定义 1 和定义 2 可知,实际发生在软件过程中的活动是数据项的收集活动,所有的度量都可由相应的数据项计算或推导出来;

(2) 在上述关于数据项的描述中,T,C 等是概念性或是指导性的,如 Tw_{ij}. T 的内容可能为"在每个过程步结束时",具体是在哪些过程步结束时要由具体的软件过程模型决定,但从逻辑上讲,Tw_{ij}. T 的内容已足以确定该数据的收集时间了.

1.2.4　GQM-D 方法的度量模型建模步骤

第 1 步. 确立度量目标;

第 2 步. 根据目标所要达到的目的、所涉及的对象实体细化出以量化的方式描述该目标的问题 Q;

第 3 步. 用度量的方式回答以上的提问;

第 4 步. 实施度量的分解;

第 5 步. 以结构化的方式描述度量及其数据项的内容.

例:依据以上建模步骤,以 SPA 的过程能力级别中的"过程执行能力"为目标的过程度量模型.

模型	说明
G 过程执行属性:该过程在运作时自觉使用了一组基础活动,且每个活动有明确的输入/输出的工作产品,并能最终满足目标的程度.	目标:考察能力 涉及实体:过程步、工作产品
Q_1 是否存在一个由活动构成的过程?	对过程步实体的提问
$M1.1$ 过程的存在度	
D_1 考察过程是否实际存在	客观数据
Q_2 是否执行了所有的基础活动?	对过程步实体的提问
$M2.1$ 活动的执行度	
$M2.1.a$ 每个过程步的执行度	对过程步的分解
D_2 考察每个过程步是否得到实际执行	客观数据
$M2.1.j$ 每个角色的执行度	对角色的分解
D_3 考察每个角色是否实际执行了自己的活动	客观数据
Q_3 每个活动是否有输入/输出产品?	对产品实体的提问
$M3.1$ 活动产品的存在度	
$M3.1.p$ 每个中间产品的存在度	对产品的分解
D_4 考察每个中间产品是否存在	客观数据
Q_4 所有的输入/输出产品是否达到了目标要求?	对产品实体的提问
$M4.1$ 活动产品的满意度	
$M4.1.p$ 每个中间产品的满意度	对产品的分解
D_5 考察每个中间产品是否达到有关标准要求	主观数据

关于 M 和 D 的结构化描述从略.

1.2.5　过程度量模型的确认准则

从前面对度量模型的研究我们知道,任何一种度量建模方法都脱离不了"经验"二字,也就是说,无论这种度量建模方法多么"深入具体"地指导了建模的过程,最后得到的模型结果都会由于建模人经验和视角的不同而产

生一定的差异.如何判断哪一个模型更合理,或者说,如何实施度量建模的质量保证是度量建模人员关心的问题.一种常用的办法是组建度量模型专家组,这在一定程度上使得产生的度量模型合理化,但还远远不够.在这些模型用于进行一定目标的度量之前,首先应当判断它是否真正度量了它想要度量的内容.这就是对度量模型所做的确认工作.

所谓度量确认准则,就是对应当存在于度量和度量目标之间的一些特定的量化关系的规定.文献[12,13]对于度量确认准则均有所讨论,IEEE Std 1061－1992 中专门设立一节推荐有关软件度量的确认准则.尽管这些准则是针对产品度量的,但由于度量所具有的共性,使得这些准则仍可用于过程度量的确认.下面关于过程度量确认的讨论是从对产品度量的确认准则中演化而来的,但却赋予了过程度量确认的、新的涵义.

定义 3(过程度量模型). 一个过程度量模型是一个树状结构 $T(G,Q,M,D)$,其中 G:根节点,表示度量目标;Q:第 2 层,表示由 G 细化而得的问题;M:第 3 层,表示对 Q 的量化回答及其在实体上的分解;D:第 4 层,表示支持 M 的计算的数据项.

度量确认准则 1. 相关性(correlation). 目的是评估 M 与 G 之间是否存在在强线性相关,以保证 M 可以成为 G 的一个间接度量值.

设 R 为 M 与 G 之间的线性相关系数,则用 R^2 表示由于 M 的变化而引起的 G 的变化,当不等式 $R^2>\beta a$ 成立时,M 与 G 具有相关性,其中 βa 为一给定的相关性阈值.

度量确认准则 2. 一致性(consistency). 目的是保证 G 的排序与 M 的排序一致,从而保证 M 可以成为 G 的一个间接度量值.

G 与 M 之间的秩相关系数 G 必须大于一给定的一致性阈值,即 $G>\beta c$. βc 为给定的一致性阈值.

度量确认准则 3. 判断能力(discriminative power). 目的是将过程能力划分为"合格和不合格"或"高能力和低能力"等判断能力,这种判断正是过程改进时必要的条件.对于一个指定的过程能力值 G_c,其相应的临界度量值 M_c 必须能够以下列方式判断所有的 G 值.

$$M_i>M_c \leftrightarrow G_i>G_c \text{ 且}$$

$$M_i=M_c \Leftrightarrow G_i=G_c.$$

该准则中的 M_c 成为衡量 G 值是否达到要求的 G_c 值的临界点,高于该临界点的 M 值,其 G 值也将高于能力目标的下限,反之,低于该临界点的 M 值将保证能够判断其相应的 G 值也低于目标能力的下限.

度量确认准则 4. 跟踪性(tracking). 目的是考察 M 是否具备跟踪 G 的变化的能力,从而判断 M 值是否可以成为 G 值的间接度量值.此时,M 的变化必须与 G 的变化一致.即对于时间 $T_1,T_2,\ldots,T_m$,以下推导成立.

$$M(T_j+1)>M(T_j) \leftrightarrow G(T_j+1)>G(T_j),$$

$$M(T_j+1)=M(T_j) \Leftrightarrow G(T_j+1)=G(T_j),$$

$$M(T_j+1)<M(T_j) \Leftrightarrow G(T_j+1)<G(T_j).$$

跟踪性使得 M 值反映了过程运作时 M 的变化对最终过程能力结果的影响,因而提供了过程改进的依据.

度量确认准则 5. 可预测性(predicability). 目的在于判断 $f(M)$ 是否能以所要求的精度预测 G 值.设 $f(M)$ 为根据 M 得到的 G 的预测值,则 $f(M)$ 必须以一定的精度逼近 G 的直接度量值 G_a,即当 $\left|\frac{G_a-f(M)}{G_a}\right|<\beta p$ 时,M 具有可预测性.

度量确认准则 6. 可重复性(repeatability). 目的是保证一个度量 M 以一定的成功率通过了对它的确认,因而可以确信该度量能够达到它的度量目的.

令 Nis 为用确认准则 i 对过程度量 M 所实施的确认的成功次数;Ni 为用确认准则 i 对过程度量 M 在所有过程模型的实例上所实施的确认的总次数;则当 $Nis/Ni>\beta is$(βis 为一认可的成功率值)时,M 具备可重复性.

以上 6 个度量确认准则是对任何一个度量提出的要求.这并不意味着所有的度量都要通过所有的确认准则.一个度量在不同的准则要求下可以表现出不同的有效性.例如,一个度量可能满足相关性准则,却不一定能满足跟踪性准则.

实际实施度量确认时经常使用统计学方法.由于篇幅所限,本文在此不予讨论.

2 模型描述机制

为了支持 GQM-D 框架中描述的度量活动，我们所使用的模型描述机制应有以下特点：(1) 直观、易理解；(2) 应可描述过程的几类实体：过程步、角色、产品、资源及约束；(3) 应能表现过程之间的时序关系. 为此，我们选择 SADT(IDEF0　ICAM DEFinition)作为我们所需的软件过程描述机制的基础.

一个典型的 SADT(IDEF0)模型是一个图和文字说明的组合，每个图有 6 个或 6 个以下的方框，用于表示过程步，每个方框如图 3 所示.

每个过程步都可以分解为另一幅 SADT(IDEF0)图及文字说明，用以描述该过程步的细化.

利用 SADT(IDEF0)的描述机制，一个数据收集类型的过程步可以表示为图 4. 其中，没有标注的箭头仅表示该过程步与其他过程步的时序关系. 进入的箭头表明该过程步在时间上偏后，出去的箭头表明该过程步在时间上偏前. 对该过程步的展开图可表示关于该数据收集步的具体收集步骤，如收集、存储等.

图 5 是在一个软件设计过程中加入过程步 $T_{W_{ij}}$ 之后的 SADT(IDEF0)图.

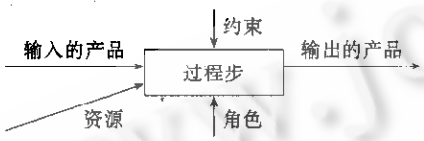


图3　SADT(IDEF0)表示的一个过程步

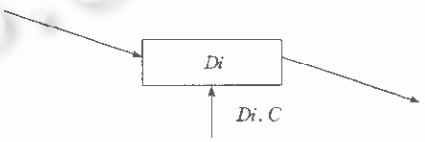


图4　SADT(IDEF0)表示的一个数据收集过程步

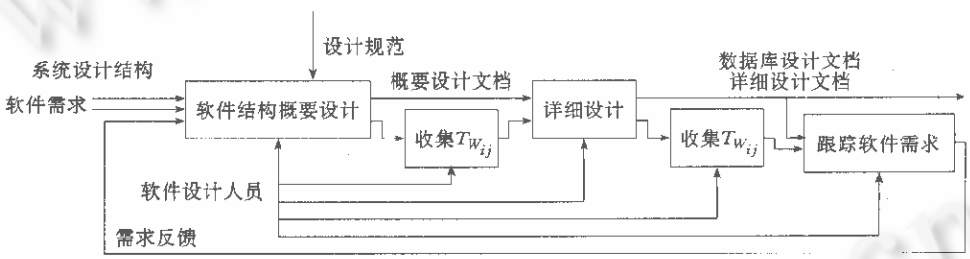


图5　增加了数据收集过程步的软件结构设计过程

3 支持过程度量的软件过程建模算法

通过以上讨论，我们有了一个 GQM-D 度量框架，该框架讨论了如何将一个度量目标细化为若干个深入到过程内部的度量，又如何描述计算这些度量的数据项，在对这些数据项的描述中包含了诸如 T——数据收集时间；C　数据收集人；P——数据收集方法等对形成数据收集过程步十分重要的信息，为在过程中创建数据收集过程步提供了充分准备；而我们讨论的模型描述机制则保证了对数据收集过程步的支持.

下面，我们给出支持过程度量的软件过程建模算法.

算法 1. 构造一个支持过程度量的过程模型.

输入：度量目标 G，被度量的软件过程 P；

输出：支持度量 G 的软件过程模型 GP-SADT；

算法：

Begin

　　根据 1.2.4 度量模型建模步骤，构造以 G 为目标的度量模型以及 M-Set 和 D-Set；

　　利用 SADT 方法构造被度量的过程 P 的描述图 P-SADT；

　　令 GP-SADT=P-SADT；

　　For each di in D-set

　　Begin

根据 $di.C$,$di.P$,$di.N$ 形成一个数据收集活动 Ai;
根据 di 创建对 Ai 的文字说明;
根据 $di.T$ 将 Ai 插入到 GP-SADT 的适合位置上;
End
End

当 M-Set=$\{Mp'\}$,D-Set=$\{T_{W_{ij}},T_{B_{ij}}\}$时,利用算法 1,也可得到如图 5 所示的"软件设计过程".

4 总 结

本文提出了一套支持过程度量的软件过程建模方法,它包括过程度量模型建模方法 GQM-D、过程描述机制 SADT(IDEF0)和支持过程度量的软件过程建模算法.其最大的优点是支持过程内部的度量,可提供过程中间度量数据,为过程改进提供关键的第一手信息.与"SEI 式"的软件过程评估方法相比,以该方法为基础进行的过程评估提供的数据更具体,更加令人信服,因而更有利于过程能力级别的改进.我们将在今后的研究工作进一步完善这一方法,并讨论以 SEI 的过程能力级别为度量目标的 GQM-D 框架、过程度量的数据收集及分析机制等,并欢迎大家就这一领域共同讨论.

参考文献

1 宿为民,黄嘉启,朱三元.关于软件过程工程概念框架的研究.计算机应用与软件,1999,16(2):1~7
(Su Wei-min, Huang Jia-qi, Zhu San-yuan. An investigation for the conception framework of software process engineering. Computer Applications and Software, 1999,16(2):1~7)

2 Paulk M C, Curtis B *et al*. Capability Maturity Model for Software. Ver1.1 SEI, CMU/SEI-93-TR-93, 1993

3 ISO/IEC/JTC1/SC7/WG10/N111-N119. Software Process Assessment (Part1-Part9), 1995

4 Bollinger Terry B, McGowan Clement. A critical look at software capability evaluations. IEEE Software, 1991,8(7):25~41

5 Pfleeger S L. Lessen learned in building a corporate metrics program. IEEE Software, 1993,10(5):67~71

6 Pfleeger S L, McGowan C. Software metrics in the process maturity framework. Journal of System and Software, 1990,12(12):255~261

7 Lott C M, Rombach H D. Measurement-based guidance of software projects using explicit project plans. Information and Software Technology, 1993,35(6,7):407~419

8 McGowan C L, Bohner S A. Model based process assessments. In: IEEE Computer Society ed. Proceedings of the 15th International Conference on Software Engineering. Los Alarnitors, CA: IEEE Computer Society Press, 1993. 202~211

9 Basili V R, Caldiera G, Rombach H D. Goal question metric paradigm. In: Marciniak J J ed. Encyclopedia of Software Engineering. New York: John Wiley & Sons, Inc., 1994. 528~532

10 DeBunje T, Saunders A. Combining process models and metrics in practice. In: Schafer W ed. Proceedings of the 4th European Workshop on Software Process Technology. Berlin: Springer-Verlag, 1995. 49~53

11 Brochers C. Differding, Günter Threin. The role of software process modeling in planning industrial measurement programs. In: IEEE Computer Society ed. Proceedings of the METRIC'96. Los Alarnitors, CA: IEEE Computer Society Press, 1996. 71~84

12 IEEE Std 1061-1992. IEEE Standard for a Software Quality Metrics Methodology, 1992

13 Methodology for Validating Software Metrics. In: Marciniak J J ed. Encyclopedia of software engineering. New York: John Wiley & Sons, Inc., 1994. 528~532

Research on Metric-supported Software Process Modeling Methodology

SU Wei-min　ZHU San-yuan

(Shanghai Development Center for Computer Software Technology　Shanghai　200050)

Abstract　In this paper, the authors suggest to think about the problem of process metric from the process modeling point of view, and discuss a methodology for metric-supported software process modeling, which includes a goal-oriented process metric modeling method, a descriptive mechanism for software process and an algorithm for metric-supported software process modeling.

Key words　Process measurement, software process modeling.