多视点需求工程中视点集成过程的研究

梁正平[1,2] 明 仲[1] 毋国庆[3]
(深圳大学信息工程学院 深圳 518060)[1] (软件工程国家重点实验室 武汉 430072)[2]
(武汉大学计算机学院 武汉 430072)[3]

摘 要 多视点需求工程方法以视点的形式,分散、独立地获取和表示不同项目相关人员的需求信息。为生成一份统一的规格说明,必须对所有相关的视点进行集成。讨论了以公共开发方式作为视点的集成方式,并基于范畴理论对集成过程进行建模。同时,以公共开发的选取为导向,提出了视点集成的两种优化策略,并证明了它们的有效性。此外,讨论了视点集成的结果与集成的顺序之间的关系。

关键词 需求工程,视点,视点集成,范畴论
中图法分类号 TP311.5 **文献标识码** A

Research on Integration Process of Viewpoints in Viewpoint-oriented Requirements Engineering

LIANG Zheng-ping[1,2] MING Zhong[1] WU Guo-qing[3]
(College of Information Engineering, Shenzhen University, Shenzhen 518060, China)[1]
(State Key Lab of Software Engineering, Wuhan 430072, China)[2] (School of Computer, Wuhan University, Wuhan 430072, China)[3]

Abstract The requirements information of all kinds of stakeholders is acquired and expressed using the form of viewpoint independently and dispersedly in Viewpoint-Oriented Requirements Engineering. It must integrate all of these viewpoints in order to yield an uniform specification. In this paper, the common development of viewpoints' specification was treated as the style of integration. The process of integration was modeled using the Category Theory. At the same time, this paper proposed two kinds of integration of viewpoints' optimizing strategies, and their validity were also proved. In addition, the relationship of the result of integration and the sequence of integration was discussed.

Keywords Requirements engineering, Viewpoint, Integration of viewpoints, Category theory

对于大型、复杂软件系统的开发,不可避免地涉及到众多的项目相关人员。由于各自背景、知识、职责等的不同,他们对待开发软件系统的看法和要求也不尽相同。为在系统开发的早期全面获取不同项目相关人员的各种需求信息,确保最终开发的软件系统能全面满足各方面用户的要求,20世纪90年代,研究人员提出了面向多视点的需求工程方法[1-3],采用视点的形式,分散、独立地获取和表示不同项目相关人员的需求信息。为生成一份完整的规格说明书,在多视点需求工程方法中,最终必须将与各视点所对应的部分规格说明集成为一个统一的整体,以作为需求工程活动结束的里程碑,以及后阶段系统开发、测试和验收的依据。

目前,在视点的定义及抽取[4,5]、视点的表示[6,7]、视点的一致性[8,9]、视点间冲突的处理[10,11]等方面已有较广泛和深入的研究。有关视点集成[12-14]的研究也不少,较有代表性的如I. Sommerville[12]和牟克典[13]等的工作。I. Sommerville的PREview方法从宏观方面指导性地提出了视点集成过程应采取的主要步骤。但他所提出的集成步骤专门针对PREview方法所采用的视点标识和表示方式而言,难以应用于集成使用其它标识和表示方式的视点。牟克典等对需求优先级以及重叠需求相对于不同视点优先级的不一致性等概念进行了严格的定义,分别采用格和乘积格来表示重叠需求相对于各视点和集成视点的优先级,借助于乘积格的准布尔特性实现对不一致优先级的容忍策略。对基于精确度量设定优先级的情形,则采用平均策略确定重叠需求相对于集成视点的综合度量,重新设定重叠需求相对于集成视点的优先级,从而解决这种不一致性。但他们对视点集成的具体方式及其过程等则缺乏进一步的讨论。

视点的集成主要包括视点冲突的检测、多种表达形式的视点间转换、视点的集成方式、集成过程等方面。主要基于范畴论[15,16]的概念和方法,从一般意义上对视点的集成方式和过程进行讨论,为多视点需求工程中视点的集成过程提供一个基础性的指南和方法。由于视点集成的方式与具体视点的内容及集成的目标相关,为统一处理视点集成时不同视点所对应部分规格说明在语法及语义方面可能存在的重叠问题,采用寻找公共开发的方式作为视点的集成方式,并将所有视点所对应部分规格说明的公共开发作为它们的集成结果。对

到稿日期:2008-09-24 返修日期:2008-12-17 本文受国家863高科技计划项目(2007AA01Z185),国家自然科学基金项目(60673122),软件工程国家重点实验室开放基金项目(SKLSE20080702),深圳大学科研启动基金项目(200747)资助。
梁正平(1979—),男,博士,副教授,主要研究领域为软件工程、形式化方法,E-mail:liangzp@szu.edu.cn;明 仲(1967—),男,博士,教授,主要研究领域为软件工程、本体论;毋国庆(1954—),男,教授,博士生导师,主要研究领域为软件工程、形式化方法。

于视点的集成过程，则以递增的方式逐步集成所有待集成的视点。同时，为便于从一般角度进一步研究与视点集成过程相关的具体性质，本文对所提出的集成过程进行了范畴建模。在此基础上，为降低集成的复杂度，提出了两种视点集成的优化策略，并对它们的有效性进行了证明。此外，对视点集成的结果与集成的顺序之间的关系等也进行了讨论。

1 视点的集成方式

实际开发过程中，若两个视点所对应的部分规格说明完全无关，则它们的集成只需将两者简单地归集到一起即可。若两个或多个视点所对应的部分规格说明存在重叠的信息，则这些重叠的信息可能语法上等价但语义上不同，也可能语法上不同但语义上等价，仅通过直接的归集操作难以将它们集成为一个完整统一的有机整体。由于在重叠信息上的具体语法及语义问题与待集成视点所对应的实际规格说明相关，因此缺乏讨论的具体对象和依据。下面主要基于开发关系[17]的方式讨论一种一般意义上的视点集成方式。

定义 1(开发关系) 需求工程过程中，若一个规格说明 S_1 可沿靠近实现的方向通过一定的方式(dv)演化为另一个规格说明 S_2，则称 S_2 是 S_1 的开发，记为 $S_2\ dv\ S_1$ 或 $S_1 \xrightarrow{dv} S_2$，同时也称 S_1 与 S_2 具有开发关系 dv。

两个规格说明 S_1 与 S_2 具有什么样的开发关系，由 S_1 的演化方式决定，且 S_2 可由 S_1 及该开发关系完全推出，故也将 S_1 与 S_2 间的开发关系 dv 称为 S_1 对应开发关系 dv。而一个规格说明具体可如何演化，即它具体可对应什么样的开发关系，与其内容及描述方式相关。

内容上，不同的规格说明与最终实现的关系不同，所应采用的演化方式也不同，故需对应不同的开发关系。例如，若 S 以不确定的方式抽象地描述待开发软件系统的行为，而 S(包括对待开发软件系统具体行为的确切定义，则与 S 对应的应为求精型或实现型的开发关系，而与 S' 对应的应为等价型的开发关系。

描述方式上，不同的描述方式所支持或定义的开发关系不同，因而实际可选取的开发关系也不同。例如，LOTOS 语言[18]中定义了路径求解、测试等价等开发关系，而 Z 语言[19]中则定义了数据求精、操作求精等开发关系。

除极个别开发关系外，由于现有各种规格说明描述方法中所定义的开发关系基本上都具有自反性和传递性，为简化讨论，下文假定所有的开发关系均满足自反性和传递性。

定义 2(公共开发) 设 $S_1, S_2, \cdots, S_n$ 为规格说明，对应的开发关系分别为 $dv_1, dv_2, \cdots, dv_n$，若存在某规格说明 S，使 $S\ dv_1\ S_1, S\ dv_2\ S_2, \cdots, S\ dv_n\ S_n$ 均成立，则称 S 为 $S_1, S_2, \cdots, S_n$ 在 $dv_1, dv_2, \cdots, dv_n$ 下的公共开发。

例：设有如下 3 个用 LOTOS 语言描述的某系统的部分规格说明，记为 VP_1_S, VP_2_S, VP_3_S，其对应的开发关系分别为 ext(扩展关系)，ext，$\leqslant_{tr}$(路径求精关系)。

$VP_1\_S :=$ *hide* $channel_1$ *in* RC_1 |[$channel_1$]| SH_1

$RC_1 :=$ *SendSMCommand*? x:*SM*; $channel_1$! x; RC_1

$SH_1 :=$ $channel_1$? x: *SM*; *SendSMtoSG*! x; *Success*; $channel_1$! x; SH_1

$VP_2\_S :=$ *hide* $channel_2$ *in* *CC* |[$channel_2$]| *CH*

CC := *CancelSMCommand*? x:*SM*; $channel_2$! x; *CC*

CH := $channel_2$? x:*SM*; *QuerySG*! x! y; *OP*(x,y)

OP(x:*SM*,y:*ST*) := ([y=*Send*]→*CancelFail*; *CH* [] [y=*Stay*]→*CancelSM*! x:*SM*; *CancelSuccess*; *CH*)

$VP_3\_S :=$ (*choice* $a \in \pi(GS_1 - \{$*SendSMCommand*, *Success*$\})$ [] a; VP_3_S) [] *SendSMCommand*? x:*SM*; *Trans*(x)

Trans(x:*SM*) := (*choice* $a \in \pi(GS_1 - \{$*SendSMCommand*, *Success*$\})$ [] a; *Trans*(x)) [] *Success*; VP_3_S

现构造规格说明 S，使 $S :=VP_1\_S$ [] VP_2_S。依据 LOTOS 中“*ext*”和“$\leqslant_{tr}$”的定义，可以验证：

S *ext* VP_1_S，S *ext* VP_2_S，$S \leqslant_{tr} VP_3\_S$

故上述的 S 是 VP_1_S, VP_2_S, VP_3_S 在 *ext*, *ext*, $\leqslant_{tr}$ 下的公共开发。

因两个或多个规格说明在各自对应的开发关系下可能存在多个公共开发，下面给出公共开发集的定义。

定义 3(公共开发集) 设 $S_1, S_2, \cdots, S_n$ 为规格说明，对应的开发关系分别为 $dv_1, dv_2, \cdots, dv_n$，它们的公共开发集 S_{CD} 定义如下：

$S_{CD} = \{S \mid S\ dv_i\ S_i$，对所有的 $i(\{1 \cdots n\})\}$

基于上面对开发关系及公共开发的定义，从语义的角度，若两个或多个视点分别对应的部分规格说明在一定的开发关系下存在公共开发，则显然该公共开发即可看成是相关视点的一个集成。

为此，视点的集成可围绕定义 1 中的开发关系 dv 展开，即以寻找各视点所对应部分规格说明的公共开发的方式作为视点的集成方式。

此外，上面所定义的公共开发也可作为定义和检查相关视点是否一致的依据。具体而言，若两个或多个视点所对应的规格说明可通过各自对应的开发关系找到一个公共开发，则说明它们都是对同一系统的合理描述，只是关注范围、抽象层次、组织形式等方面可能不同，因而是一致的。相反，若两个或多个不同的规格说明不存在一个公共的开发，则说明它们相互间存在某些不能协调的东西，无法找到一个公共的开发使各规格说明中的描述能有机地融合到一起，因而是不一致的。

定义 4(视点一致) 设 $S_1, S_2, \cdots, S_n$ 分别为视点 $VP_1, VP_2, \cdots, VP_n$ 所对应的部分规格说明，且所对应的开发关系分别为 $dv_1, dv_2, \cdots, dv_n$，若存在某规格说明 S，使 $S\ dv_1\ S_1, S\ dv_2\ S_2, \cdots, S\ dv_n\ S_n$ 均成立，则称 $VP_1, VP_2, \cdots, VP_n$ 一致。

2 视点的集成过程及其范畴建模

2.1 集成过程

基于上面介绍的集成方式，视点集成的目标是找出一个所有待集成视点各自所对应部分规格说明的公共开发。但当存在多个视点需进行集成时，一方面想一次性找出所有部分规格说明的公共开发可能难度较大，另一方面从一致性检查的角度而言，若它们之间存在不一致，期望采用一次性方式找出它们的公共开发，也不便于精确地定位具体哪些视点之间存在不一致，故视点的集成必须采用逐步的方式进行。

逐步集成可分为无序集成和递增集成两种形式。前者每次从暂未参与集成的规格说明和前面阶段通过集成生成的中

间结果中随机选取不确定的若干个进行集成，生成新的中间集成结果，直至找到所有规格说明的一个公共开发；后者除第一步外，每次将前一步生成的中间集成结果与一个或多个暂未参与集成的规格说明进行集成，生成新的中间集成结果，通过递增的方式，当所有规格说明均参与集成时，则生成的结果即为所有规格说明的公共开发。

无序集成与递增集成本质上没有什么区别，但后者的集成过程更加清晰、有序。不失一般性，下面讨论基于二元递增方式的逐步集成方法，如图 1 所示。

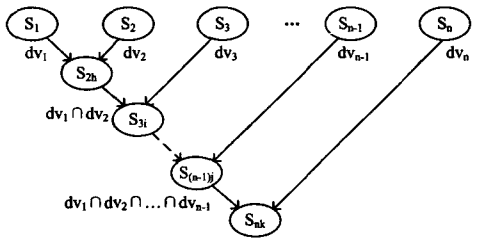

图 1　基于二元递增方式的视点集成

图中，$S_1, S_2, \cdots, S_n$ 分别表示视点 $VP_1, VP_2, \cdots, VP_n$ 所对应的部分规格说明，且所对应的开发关系分别为 $dv_1, dv_2, \cdots, dv_n$，S_{2h} 为 S_1 和 S_2 的公共开发，S_{3i} 为 S_{2h} 和 S_3 的公共开发，依此类推，S_{nk} 为 $S_{(n-1)j}$ 和 S_n 的公共开发。

由于采用寻找公共开发的方式进行视点的集成，为使每次生成的中间集成结果可进一步参与后续的集成，故它们也必须对应一个合理、有效的开发关系。在此，指定每个中间集成结果所对应的开发关系为其前一个中间集成结果及参与此次集成的规格说明两者所分别对应的开发关系的合取。即如图 1 所示，S_{2h} 对应的开发关系为 $dv_1 \cap dv_2$，S_{3i} 对应的开发关系为 $dv_1 \cap dv_2 \cap dv_3$。依此类推，$S_{(n-1)j}$ 对应的开发关系为 $dv_1 \cap dv_2 \cap \cdots \cap dv_{n-1}$。这样，若 S_{3i} 是 S_{2h} 和 S_3 的公共开发，则 S_{3i} 与 S_{2h} 之间既满足 dv_1 关系，又满足 dv_2 关系，同时 S_{2h} 与 S_1, S_2 间分别满足 dv_1, dv_2 关系。因假设所有的开发关系均为偏序，故 S_{3i} 也是 S_1 和 S_2 的公共开发，依此类推，S_{nk} 是所有 $S_1, S_2, \cdots, S_n$ 的公共开发。

基于二元递增方式的视点集成形式上虽然简单，但实际的操作过程却可能十分繁琐。原因在于两个规格说明的公共开发集中可能包括多个元素，其中有些元素可能与下一个待集成的规格说明进行进一步集成，另一些则可能无法进一步集成，亦即无法找出下一个所需的新的公共开发。故基于二元递增方式集成规格说明时，必须恰当地选取中间过程所生成的公共开发，若某些选取的公共开发导致后面的过程无法完成最终的集成，则必须选取其它的公共开发进行进一步的集成。

综上，下面以算法的形式对基于二元递增方式的集成过程进行具体描述。

设有 n 个视点需进行集成，各自所对应的规格说明分别为 $S_1, S_2, \cdots, S_n$，且这些规格说明所对应的开发关系分别为 $dv_1, dv_2, \cdots, dv_n$。不失一般性，假设按 $S_1, S_2, \cdots, S_n$ 的排列顺序依次集成。

算法 1　基于二元递增方式的视点集成过程

步骤 1　计算 S_1, S_2 在 dv_1, dv_2 下的公共开发集 S_{CD_1}；

步骤 2　从 S_{CD_1} 中任取一个，记为 S_{2h}，同时 $S_{CD\_1} = S_{CD\_1} - \{S_{2h}\}$，计算 S_{2h}, S_3 在 $dv_1 \cap dv_2, dv_3$ 下的公共开发集 S_{CD_2}；

步骤 3　若 S_{CD_2} 为空，则转前一步，否则转下一步；

步骤 4　若 S_{CD_2} 为空，则转前两步，否则从 S_{CD_2} 中任取一个，记为 S_{3i}，同时 $S_{CD\_2} = S_{CD\_2} - \{S_{3i}\}$，计算 S_{3i}, S_4 在 $dv_1 \cap dv_2 \cap dv_3, dv_4$ 下的公共开发集 S_{CD_3}；

……

依此类推

……

步骤 $2n-5$　若 S_{CD_n-2} 为空，则转前一步，否则转下一步；

步骤 $2n-4$　若 S_{CD_n-2} 为空，则转前两步，否则从 S_{CD_n-2} 中任取一个，记为 $S_{(n-1)j}$，同时 $S_{CD\_n-2} = S_{CD\_n-2} - \{S_{(n-1)j}\}$，计算 $S_{(n-1)j}, S_n$ 在 $dv_1 \cap dv_2 \cap \cdots \cap dv_{n-1}, dv_n$ 下的公共开发集 S_{CD_n-1}；

步骤 $2n-3$　若 S_{CD_n-1} 为空，则转前一步，否则 S_{CD_n-1} 中的任意一个均为 $S_1, S_2, \cdots, S_n$ 在 $dv_1, dv_2, \cdots, dv_n$ 下的公共开发，即 $S_1, S_2, \cdots, S_n$ 的集成。

基于算法 1 所示的集成过程及定义 4 对视点一致的定义，显然存在如下的定理。

定理 1　$n(n \geqslant 0)$个视点一致，当且仅当算法 1 能实现对它们的集成。

定理 1 的充分性和必要性可分别根据定义 4 和算法 1 进行证明，限于篇幅，本文不进行具体介绍。

2.2　集成过程的范畴建模

由于本文主要从一般意义上基于规格说明间的开发关系研究视点的集成，故对视点集成过程的研究也主要关注于规格说明间存在的开发关系，而非各视点的具体内容及其表示形式。与集合论以个体的组成元素为研究对象不同，范畴论关注于研究对象间的态射，即对象间存在的关系，通过对对象间态射的研究，从最一般的角度分析对象及其相互之间可能具有的性质。基于本文所采用的集成方式及范畴论的这一特点，为便于从一般意义上研究上一节介绍的视点集成过程，下面采用范畴论的方法对该过程进行建模。

设有 n 个视点需进行集成，各自所对应的部分规格说明分别为 $S_1, S_2, \cdots, S_n$，且这些规格说明所对应的开发关系分别为 $dv_1, dv_2, \cdots, dv_n$。假设按 $S_1, S_2, \cdots, S_n$ 的排列顺序以二元递增方式依次集成，则该集成过程可用如下构造的范畴 A 进行建模：

(1) 将 $S_1, S_2, \cdots, S_n$ 看成是 A 的对象，且分别具有自身到自身的态射 $dv_1, dv_2, \cdots, dv_n$；

(2) 假设 S_1, S_2 的公共开发集为 S_{CD_1}，将 S_{CD_1} 中的每个元素看成是 A 的对象，且它们均具有自身到自身的态射 $dv_1 \cap dv_2$；

(3) 分别在 S_1, S_2 与 S_{CD_1} 的每个元素间建立态射 dv_1, dv_2；若 S_{CD_1} 中的两个元素间具有 $dv_1 \cap dv_2$ 关系，则在它们之间建立相应的态射 $dv_1 \cap dv_2$；

(4) 假设 S_{CD_1} 中的每个元素与 S_3 的所有公共开发集中的元素构成集合 S_{CD_2}，将 S_{CD_2} 中的每个元素看成是 A 的对象，且它们均具有自身到自身的态射 $dv_1 \cap dv_2 \cap dv_3$；

(5) 在 S_{CD_1} 与 S_{CD_2} 的元素两两间建立态射 $dv_1 \cap dv_2$，在 S_3 与 S_{CD_2} 的每个元素间建立态射 dv_3；若 S_{CD_2} 中的两个元素间具有 $dv_1 \cap dv_2 \cap dv_3$ 关系，则在它们之间建立相应的态射

$dv_1 \bigcap dv_2 \bigcap dv_3$；

……

依此类推

……

(2n−2) 假设 S_{CD_n-2} 中的每个元素与 S_{n-1} 的所有公共开发集中的元素构成集合 S_{CD_n-1}，将 S_{CD_n} 中的每个元素看成是 A 的对象，且它们均具有自身到自身的态射 $dv_1 \bigcap dv_2 \bigcap \cdots \bigcap dv_n$；

(2n−1) 在 S_{CD_n-2} 与 S_{CD_n-1} 的元素两两间建立态射 $dv_1 \bigcap dv_2 \bigcap \cdots \bigcap dv_{n-1}$，在 S_{n-1} 与 S_{CD_n-1} 的每个元素间建立态射 dv_n；若 S_{CD_n-1} 中的两个元素间具有 $dv_1 \bigcap dv_2 \bigcap \cdots \bigcap dv_n$ 关系，则在它们之间建立相应的态射 $dv_1 \bigcap dv_2 \bigcap \cdots \bigcap dv_n$；

(2n) 除经上述态射的合成生成的新态射外，A 中不包含任何其它的对象和态射。

其中，A 中态射的合成操作符“∘”定义如下：

设 A，B，C 为 A 中的对象，$dv_{x1}: A \times B$，$dv_{x2}: B \times C$ 为 A 中的态射，则 $dv_{x2} \circ dv_{x1}: A \times C$ 定义为：

$dv_{x2} \circ dv_{x1} \equiv dv_{x1}$

因所有开发关系 $dv_1, dv_2, \cdots, dv_n$ 均满足自反性和传递性，容易验证上面(1)到(2n−1)的构造过程是合理和有效的，故它们是对算法 1 的正确建模。下面证明由(1)到(2n)及合成操作符“∘”所定义的 A 确实构成一个范畴。

首先讨论合成操作符“∘”定义的有效性。

在上面(1)到(2n−1)的构造过程中，对于任意的对象 A 和 B，若它们之间存在态射 $dv_{x1}: A \times B$，则该态射在意义上表示 $B\ dv_{x1}\ A$。故为使 $dv_{x2} \circ dv_{x1} \equiv dv_{x1}$ 有效，也必须保证 $C\ dv_{x1}\ A$ 成立。

设 $dv_{x1} = dv_i \bigcap \cdots \bigcap dv_j$，$(1 \leqslant i \leqslant j \leqslant n)$，$dv_{x2} = dv_l \bigcap \cdots \bigcap dv_m$，$(1 \leqslant l \leqslant m \leqslant n)$，

$|dv_{x1}| \equiv \{dv_i, \cdots, dv_j\}$，$|dv_{x2}| \equiv \{dv_l, \cdots, dv_m\}$

根据(1)到(2n−1)的构造过程可知：$|dv_{x1}| \subseteq |dv_{x2}|$，又由于 $dv_i (1 \leqslant i \leqslant n)$ 均具有传递性，故若 $B\ dv_{x1}\ A$，$C\ dv_{x2}\ B$ 成立，则 $C\ dv_{x1}\ A$ 必然成立。

因此，上面合成操作符“∘”的定义有效。

下面讨论 A 满足态射乘积的存在性、态射乘积的结合性以及恒等态射的存在性。

根据对 A 中对象，态射及态射合成操作符“∘”定义，容易验证：

i) $\forall dv_{x1}, dv_{x2} \in \mathrm{Mor}(A)$，$dv_{x2} \circ dv_{x1} \in \mathrm{Mor}(A)$；

ii) $\forall dv_{x1} \in \mathrm{Mor}(A, B)$，$dv_{x2} \in \mathrm{Mor}(B, C)$，$dv_{x3} \in \mathrm{Mor}(C, D)$，

$dv_{x3} \circ (dv_{x2} \circ dv_{x1}) = (dv_{x3} \circ dv_{x2}) \circ dv_{x1}$；

iii) $\forall A \in \mathrm{Ob}(A)$，均存在唯一一个自身到自身的态射，此处假设用 id_A 统一表示，且对于 $\forall dv_{x1} \in \mathrm{Mor}(A, B)$，$dv_{x2} \in \mathrm{Mor}(C, A)$，

$dv_{x1} \circ id_A = dv_{x1}$，$id_A \circ dv_{x2} = dv_{x2}$

其中 i)，ii)，iii)分别表示态射乘积的存在性、态射乘积的结合性和恒等态射的存在性，故 A 满足范畴的定义[15,16]，因而 A 确实构成一个范畴。

要注意的是，在上面的 A 中，对于其中任意一个对象，从该对象出发的态射均采用相同的名称进行标识。

由于 A 是对算法 1 的范畴建模，根据性质 1 及 A 的构造过程，若 $S_1, S_2, \cdots, S_n$ 在 $dv_1, dv_2, \cdots, dv_n$ 下一致，则 A 中必然至少存在一个对象 S，使 $S_1, S_2, \cdots, S_n$ 与 S 间分别存在态射 $dv_1, dv_2, \cdots, dv_n$，该 S 即为 $S_1, S_2, \cdots, S_n$ 在 $dv_1, dv_2, \cdots, dv_n$ 下的公共开发。反之，则不存在这样的对象 S。此外，A 中的对象 $S_1, S_2, \cdots, S_n$ 自身到自身分别只存在一个唯一的态射 $dv_1, dv_2, \cdots, dv_n$，而 $dv_i \circ dv_i = dv_i (1 \leqslant i \leqslant n)$。故根据范畴论中对共锥的定义[15,16]可知，$\{dv_i: S_i \to S\} (1 \leqslant i \leqslant n)$ 是 A 中由 $S_1, S_2, \cdots, S_n$ 所构成离散图的共锥。

定义 5(共锥元) 设 $A_1, A_2, \cdots, A_m$ 为 A 中的对象，若 $\{dv_{xi}: A_i \to B\} (1 \leqslant i \leqslant m)$ 为离散图 $A_1, A_2, \cdots, A_m$ 在 A 中的共锥，则称 B 为 $A_1, A_2, \cdots, A_m$ 在 A 中的共锥元。$A_1, A_2, \cdots, A_m$ 所有共锥元的集合记作 $Cocone\_E(A_1, A_2, \cdots, A_m)$。

设从 $A_1, A_2, \cdots, A_m$ 出发的态射分别标记为 $dv_{x1}, dv_{x2}, \cdots, dv_{xm}$，则根据 A 中态射的意义，显然有 $Cocone\_E(A_1, A_2, \cdots, A_m)$ 等价于 $S_{CD}(dv_{x1}, A_1)(dv_{x2}, A_2) \cdots (dv_{xm}, A_m)$，故对于规格说明的一致性及其集成的性质，也可用如下的范畴论术语描述。

性质 1 (i)若 A 中由 $S_1, S_2, \cdots, S_n$ 所构成的离散图存在共锥，则它们相互一致，反之则不一致；(ii)若 A 中由 $S_1, S_2, \cdots, S_n$ 所构成的离散图存在共锥，则任意一个共锥元均可作为它们的一个集成。

3 视点集成过程的优化策略

在 2.1 节中已经讨论了若中间过程的公共开发选取不当，可能造成对客观上一致的视点也无法进行下一步的集成。为此，算法 1 采用回溯的方式逐个尝试中间过程的公共开发集中所有可能的公共开发，直至完成全部视点的最终集成。但在实际开发过程中，每个公共开发集中可能包括多个甚至无穷多个公共开发。最坏情况下，算法 1 的时间复杂度与待集成视点的个数及每个公共开发集中元素的个数均呈指数关系，其计算量相当巨大，故必须考虑采用有效的策略降低集成过程的复杂性。

理想情况下，当所有待集成视点均一致时，若每次均能选取最为恰当的一个公共开发，则 n 个视点的集成只需计算 n−1 次公共开发即可。为此，下面围绕公共开发集中公共开发的选取策略进行讨论。

定义 6(共极限元) 设 $A_1, A_2, \cdots, A_m$ 为 A 中的对象，$\{dv_{xi}: A_i \to B\} (1 \leqslant i \leqslant m)$ 为离散图 $A_1, A_2, \cdots, A_m$ 在 A 中的共极限，则称 B 为 $A_1, A_2, \cdots, A_m$ 在 A 中的共极限元。

由于范畴的共极限在同构的意义上唯一，故若 $A_1, A_2, \cdots, A_m$ 在 A 中存在共极限元，则其共极限元也在同构的意义上唯一。将 $A_1, A_2, \cdots, A_m$ 的共极限元记作 $Colimit\_E(A_1, A_2, \cdots, A_m)$。

根据定义 5 和定义 6，$Colimit\_E(A, B)$ 是 $Cocone\_E(A, B)$ 中一个在同构意义上唯一的元素，因此集成的最佳可能策略是每次首先选取相关的共极限元进行集成。但对于 A 中的两个对象 A、B，$Colimit\_E(A, B)$ 不一定存在。下面首先考虑一种条件稍宽的集成策略。

定义 7(极大共锥元) 设 $A_1, A_2, \cdots, A_m$ 为 A 中的对象，从它们出发的态射分别标记为 $dv_{x1}, dv_{x2}, \cdots, dv_{xm}$，C 属于 $Cocone\_E(A_1, A_2, \cdots, A_m)$，若对于 $Cocone\_E(A_1, A_2, \cdots, A_m)$ 中任意不与 C 同构的对象 C′，C′ 到 C 间不存在标记为 $dv_{x1} \bigcap$

$dv_{x2} \cap \cdots \cap dv_{xm}$ 的态射，则称 C 为 $A_1, A_2, \cdots, A_m$ 在 A 中的极大共锥元。$A_1, A_2, \cdots, A_m$ 所有极大共锥元的集合记作 $Cocone\_M(A_1, A_2, \cdots, A_m)$。

由于极大共锥元的个数一般比共锥元的个数少很多，且极大共锥元通常存在，故可考虑在算法 1 中不依次求取并处理共锥元集，而仅依次求取并处理相应的极大共锥元集。

策略 1 在算法 1 中不依次求取并处理共锥元集 $Cocone\_E_1, Cocone\_E_2, \cdots, Cocone\_E_{n-1}$，而仅依次求取并处理相应的极大共锥元集 $Cocone\_M_1, Cocone\_M_2, \cdots, Cocone\_M_{n-1}$。

下面讨论策略 1 的有效性。

假设 1 设 $A_1, A_2, \cdots, A_m$ 为 A 中的任意对象，从它们出发的态射分别标记为 $dv_{x1}, dv_{x2}, \cdots, dv_{xm}$，若 $Cocone\_E(A_1, A_2, \cdots, A_m)$ 不为空，则对任意的 $C(Cocone\_E(A_1, A_2, \cdots, A_m)$，存在一个 $C(Cocone\_M(A_1, A_2, \cdots, A_m)$，使 C' 到 C 存在标记为 $dv_{x1} \cap dv_{x2} \cap \cdots \cap dv_{xm}$ 的态射。

根据定义 7，若 A 中两个待集成对象的所有共锥元间不构成无穷链，或对于每条可能的无穷链，总存在某个非无穷链上的对象到该无穷链上所有对象的态射，则假设 1 显然成立。要注意的是，若两个对象的共锥元间通过相应态射构成一个回路，根据范畴论知识，该回路上的对象在同构的意义上等价，故可将它们看作是同一个对象，因而与无穷链问题无关。由于现有各种规格说明描述方法所定义的开发关系基本上均满足假设 1 的要求，故假设 1 在通常情况下是可行和有效的。

性质 2 设 A_i', A_j 为规格说明，dv_{xi}', dv_{xj} 为开发关系，其中 $1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n$，若 $\{(dv_{x1}', A_1'), \cdots, (dv_{xm}', A_n')\} \subseteq \{(dv_{x1}, A_1), \cdots, (dv_{xm}, A_n)\}$，则 $S_{CD}(dv_{x1}, A_1) \cdots (dv_{xm}, A_n) \subseteq S_{CD}(dv_{x1}', A_1') \cdots (dv_{xm}', A_m')$。

性质 2 可从定义 3 直接得证，从略。由性质 2 可直接得到如下的性质。

性质 3 设 A_i', A_j 为范畴 A 中的对象，其中 $1 \leqslant i \leqslant m$，$1 \leqslant j \leqslant n$，若 $\{A_1', \cdots, A_m'\} \subseteq \{A_1, \cdots, A_n\}$，则 $Cocone\_E(A_1, \cdots, A_n) \subseteq Cocone\_E(A_1', \cdots, A_m')$。

定理 2 若相关规格说明一致，且假设 1 成立，则策略 1 能有效地对它们进行集成。

分析：假设对 n 个需求规格说明 $S_1, S_2, \cdots, S_n$ 进行集成，它们对应的开发关系分别为 $dv_1, dv_2, \cdots, dv_n$。策略 1 有效是指当 $\overset{1..n}{C}(dv_i, S_i)$ 成立时，总存在非空的 $Cocone\_M_1, Cocone\_M_2, \cdots, Cocone\_M_{n-1}$。

下面采用归纳法证明。

证明：

(1)当 $m=2$ 时，由于 $C(dv_1, S_1)(dv_2, S_2)$ 成立，故 $Cocone\_E(S_1, S_2)$ 不为空。

又由于假设 1 成立，显然可知 $Cocone\_M(S_1, S_2)$ 不为空，即 $Cocone\_M_1$ 不为空。

当 $m=3$ 时，

由于 $C(dv_1, S_1)(dv_2, S_2)(dv_3, S_3)$ 成立，故 $Cocone\_E(S_1, S_2, S_3)$ 不为空。

设 $S \in Cocone\_E(S_1, S_2, S_3)$，显然 $S\ dv_3\ S_3$ ①

由性质 3 可知：$S \in Cocone\_E(S_1, S_2)$，又假设 1 成立，故必然存在某个 $S_1' \in Cocone\_M(S_1, S_2)$，即存在某个 $S_1' \in Cocone\_M_1$，且 $S\ dv_1 \cap dv_2\ S_1'$ ②

由①，②可知：$S \in Cocone\_E(S_1', S_3)$。

再由假设 1 可知：必然存在某个 $S_2' \in Cocone\_E(S_1', S_3)$，且 $S\ (dv_1 \cap dv_2) \cap dv_3\ S_2'$，因而 $Cocone\_M_2$ 不为空。

(2)假设 $m=k$ 时命题成立。

由于 $\overset{1..k+1}{C}(dv_i, S_i)$ 成立，故 $Cocone\_E(S_1, S_2, \cdots, S_{k+1})$ 不为空，设 $S \in Cocone\_E(S_1, S_2, \cdots, S_{k+1})$，

显然：$S\ dv_{k+1}\ S_{k+1}$ ③

由性质 3 可知：

$S \in Cocone\_E(S_1, S_2, \cdots, S_k), \cdots, S \in Cocone\_E(S_1, S_2)$

又由于 $m=k$ 时命题成立，故存在 $S_1' \in Cocone\_M_1$，且 $S\ dv_1 \cap dv_2\ S_1'$

$S_2' \in Cocone\_M_2$，且 $S\ (dv_1 \cap dv_2) \cap dv_3\ S_2'$，

……

$S_{k-1}' \in Cocone\_M_{k-1}$，且 $S(dv_1 \cap \cdots \cap dv_{k-1}) \cap dv_k S_{k-1}'$ ④

由③，④可知：$S \in Cocone\_E(S_{k-1}', S_{k+1})$。

再由假设 1 可知：存在某个 $S_k' \in Cocone\_E(S_{k-1}', S_{k+1})$，且 $S(dv_1 \cap \cdots \cap dv_k) \cap dv_{k+1}\ S_k'$，因而 $Cocone\_M_k$ 不为空。

综上(1)，(2)，定理 2 得证。

下面讨论与共极限元相关的集成策略。

由于两个对象的所有共极限元在同构的意义下唯一，因此若相应的共极限元存在，则可考虑在算法 1 中仅依次求取并处理共极限元。若能达到集成的目的，则显然该集成方式的复杂度最低。下面讨论该策略的有效性。

假设 2 设 $A_1, A_2, \cdots, A_m$ 为 A 中的任意对象，若 $Cocone\_E(A_1, A_2, \cdots, A_m)$ 不为空，则 $Colimit\_E(A_1, A_2, \cdots, A_m)$ 存在。

设 A 中从 A, B 出发的态射分别标记为 dv_A, dv_B，根据范畴论中共极限与共锥间的关系，显然在 A 中存在从 $Colimit\_E(A, B)$ 到 $Cocone\_E(A, B)$ 中所有对象的标记为 $dv_A \cap dv_B$ 的态射。

策略 2 在算法 1 中不依次求取并处理共锥元集 $Cocone\_E_1, Cocone\_E_2, \cdots, Cocone\_E_{n-1}$，而仅依次求取并处理相应的共极限元 $Colimit\_E_1, Colimit\_E_2, \cdots, Colimit\_E_{n-1}$。

定理 3 若相关规格说明一致，且假设 2 成立，则策略 2 能有效地对它们进行集成。

定理 3 的证明与定理 2 类似，从略。

性质 4 若假设 2 成立，A, B, C 为 A 中的对象，从它们出发的态射分别标记为 dv_A, dv_B, dv_C，则在 A 中存在从 $Colimit\_E(A, B)$ 到 $Colimit\_E(A, B, C)$ 标记为 $dv_A \cap dv_B$ 的态射。

证明：由范畴论中共极限与共锥的关系，显然 $Colimit\_E(A, B, C) \in Cocone\_E(A, B, C)$。再据性质 3，可知：$Colimit\_E(A, B, C) \in Cocone\_E(A, B)$，故由范畴 A 中态射的构造方式，可知 A 中存在从 $Colimit\_E(A, B)$ 到 $Colimit\_E(A, B, C)$ 标记为 $dv_A \cap dv_B$ 的态射。

定理 4 若假设 2 成立，A, B, C 为 A 中的对象，则 $Colimit\_E(A, Colimit\_E(B, C))$ 与 $Colimit\_E(A, B, C)$ 在 A 中等价。

证明：设从 A, B, C 出发的态射分别标记为 dv_A, dv_B, dv_C，

由范畴论中共极限与共锥的关系，可知 $Colimit\_E(A,$

$Colimit\_E(B,C)) \in Cocone\_E(A,B,C)$。

而 $Colimit\_E(A,B,C)$ 为 A,B,C 的共极限元，故据 A 中态射的构造方式，可知 A 中存在从 $Colimit\_E(A,B,C)$ 到 $Colimit\_E(A,Colimit\_E(B,C))$ 标记为 $dv_A \bigcap dv_B \bigcap dv_C$ 的态射，记作 f。

此外，据性质 4，可知：A 中存在从 $Colimit\_E(B,C)$ 到 $Colimit\_E(A,B,C)$ 标记为 $dv_B \bigcap dv_C$ 的态射；据定义 6，A 中存在从 $Colimit\_E(A,B,C)$ 到 A 的标记为 dv_A 的态射，故据定义 5 及 A 中态射的构造方式，可知

$$Colimit\_E(A,B,C) \in Cocone\_E(A,Colimit\_E(B,C))$$

而 $Colimit\_E(A,Colimit\_E(B,C))$ 为 A 和 $Colimit\_E(B,C)$ 的共极限元，故同样据 A 中态射的构造方式，可知 A 中存在从 $Colimit\_E(A,Colimit\_E(B,C))$ 到 $Colimit\_E(A,B,C)$ 标记为 $dv_A \bigcap dv_B \bigcap dv_C$ 的态射，记作 g。

又据 A 中恒等态射的构造方式，可知 $id_{Colimit\_E(A,B,C)}$ 和 $id_{Colimit\_E(A,Colimit\_E(B,C))}$ 均标记为 $dv_A \bigcap dv_B \bigcap dv_C$。

再据 A 中态射合成操作符“$\circ$”的定义，可知：

$$f \circ g = (dv_A \bigcap dv_B \bigcap dv_C) \circ (dv_A \bigcap dv_B \bigcap dv_C) = dv_A \bigcap dv_B \bigcap dv_C = id_{Colimit\_E(A,B,C)}$$

$$g \circ f = (dv_A \bigcap dv_B \bigcap dv_C) \circ (dv_A \bigcap dv_B \bigcap dv_C) = dv_A \bigcap dv_B \bigcap dv_C = id_{Colimit\_E(A,Colimit\_E(B,C))}$$

故 $Colimit\_E(A,Colimit\_E(B,C))$ 与 $Colimit\_E(A,B,C)$ 在 A 中等价。

定理 5 若假设 2 成立，A,B,C 为 A 中的对象，则 $Colimit\_E(Colimit\_E(A,B),C)$ 与 $Colimit\_E(A,B,C)$ 在 A 中等价。

定理 5 的证明与定理 4 类似，从略。

定理 6 若假设 2 成立，A,B,C 为 A 中的对象，则 $Colimit\_E(A,Colimit\_E(B,C))$ 与 $Colimit\_E(Colimit\_E(A,B),C)$ 在 A 中等价。

从定理 4 和定理 5 可直接得到定理 6 的证明。

定理 6 表明，在假设 2 成立的情况下，对于 A 中的任意 3 个对象，不论采用何种顺序求取它们的共极限元，最终的结果都是等价的，即在同构的意义下唯一。因此，若假设 2 成立，当采用策略 2 对 n 个规格说明进行集成时，不论采用何种集成顺序，它们的集成结果均相互等价，即在同构的意义下唯一，换言之，集成的结果与集成的顺序无关。

基于范畴 A 的构造，上面通过对策略 1 及策略 2 有效性的讨论，证明了在集成过程中只需对共锥元集的某一特定子集进行处理，则既可有效地对规格说明进行集成，又可减少集成过程中回溯的次数，从而实现对算法 1 的优化。

此外，若范畴 A 是共完全的，即由 A 中任意对象和态射组成的图在 A 中均存在共极限，则显然与整个 A 对应的图 Graph——即 A 中所有的对象和态射均与 Graph 的相应顶点及边一一对应——在 A 中存在共极限，设为 $Colimit\_M_{Graph}$。根据 A 的构造方式及 A 中态射的意义，显然 $Colimit\_M_{Graph}$ 为 A 中所有对象在对应开发关系（即从各对象到 $Colimit\_M_{Graph}$ 的态射）下的公共开发。因此，当采用算法 1 对相关视点进行集成时，在集成过程的每个步骤，不论选取相应公共开发集中的哪个公共开发，该公共开发均可使随后所有的集成操作顺利进行，而不需任何回溯，即有如下的性质。

性质 5 若范畴 A 共完全，则在执行算法 1 时不需进行任何回溯操作。

性质 5 可根据范畴论中共完全的定义直接进行证明，从略。

由性质 5，当范畴 A 共完全时，由于算法 1 中不需进行任何回溯操作，显然此时 n 个视点的集成只需直接计算 $n-1$ 次公共开发。

结束语 视点的集成是多视点需求工程方法中极其重要也是无法回避的问题。针对与视点集成相关的问题，本文从集成方式、集成过程、集成过程的建模及其优化等方面对视点的集成进行了讨论。

对于视点的集成方式，本文基于规格说明间开发关系的定义，将所有待集成视点所对应部分规格说明的公共开发作为视点的集成目标，以寻找公共开发的方式进行视点的集成。对于视点的集成过程，不失一般性，本文主要讨论了一种基于二元递增方式的逐步集成方法，通过依次将待集成视点与所生成的中间集成结果进行集成，最终实现对所有视点的集成。

为便于研究该集成过程的相关性质及对其进行优化，本文对该集成过程进行了范畴建模，并证明了所构造的模型满足范畴的定义。同时，为降低集成过程的复杂性，以公共开发集中公共开发的选取为导向，本文基于视点集成过程的范畴模型提出了视点集成的两种优化策略：第一种优化策略每次仅针对公共开发集中的极大共锥元集进行集成；第二种优化策略则进一步缩小公共开发的选取范围，将每次选取的公共开发限定为共极限元。在相应前提条件成立的情况下，这两种优化策略能有效降低集成过程的复杂度。

此外，本文证明了在所构建的范畴中，若当任意对象集合的共锥元集不为空时，对应的共极限元存在，则视点集成的结果与集成的顺序无关。特别地，若该范畴满足共完全，则集成过程中不论选取哪个公共开发，均可使随后的集成操作能顺利进行，而不需任何回溯。

参 考 文 献

[1] Nuseibeh B, Kramer J, Finkelstein A. A Framework for Expression the Relationships Between Multiple Views in Requirements Specification [J]. IEEE Transaction on Software Engineering, 1994, 20(10): 760-773

[2] Kotonya G, Sommerville I. Requirements Engineering with Viewpoints [J]. Software Engineering Journal, 1996, 11(1): 5-18

[3] Finkelstein A, Sommerville I. The Viewpoints FAQ [J]. Software Engineering Journal, 1996, 11(1): 2-4

[4] Kaiya H, Saeki M. Weaving multiple viewpoint specifications in goal oriented requirements analysis [A]// Proceedings of the 2004 Asia-Pacific Software Engineering Conference [C]. Los Alamitos: IEEE CS Press, 2004: 418-427

[5] Aoyama K, Ugai T, Yamada S, et al. Extraction of Viewpoints for Eliciting Customer's Requirements based on Analysis of Specification Change Records [A]// Proceedings of the 2007 Asia-Pacific Software Engineering Conference [C]. Los Alamitos: IEEE CS Press, 2007: 33-44

[6] Silva A. Requirements, Domain and Specifications: A Viewpoint-based Approach to Requirements Engineering [A]// Proceeding of the 24th International Conference on Software Engineering [C]. Los Alamitos: IEEE CS Press, 2002: 94-104

[7] 梁正平，毋国庆，王志强. 一种基于问题框架的视点表示模型

[J]. 计算机工程,2007,33(15):67-69

[8] Nentwich C, Emmerich W, Finkelstein A, et al. Flexible Consistency Checking [J]. ACM Transactions on Software Engineering and Methodology, 2003, 12(1): 28-63

[9] Liang Z P, Wu G Q. Consistency checking of multiviews based on agent [C]// Proceeding of the 2004 International Conference on Computer and Information Technology. Los Alamitos: IEEE CS Press, 2004: 1087-1091

[10] Nuseibeh B, Kramer J, Finkelstein A. Viewpoints: Meaningful Relationships are Difficult [C]// Proceeding of the 25th International Conference on Software Engineering. Los Alamitos: IEEE CS Press, 2003: 676-681

[11] 江敏,毋国庆. 多视点间不一致性的认知推理[J]. 小型微型计算机系统,2007,28(2):322-327

[12] Sommerville I, Sawyer P. Requirements Engineering: A Good Practice Guide [M]. Chichester: John Wiley & Sons, 1997

[13] 牟克典,金芝,陆汝钤. 视点合成中重叠需求的不一致优先级处理[J]. 计算机学报,2004,27(10):1379-1387

[14] Ghose A, Qiuming L. Viewpoints Merging via Incrementally Elicited Ranked Structures [C]// Proceeding of the 6th International Conference on Quality Software. Los Alamitos: IEEE CS Press, 2006: 141-150

[15] Pierce B C. Basic Category Theory for Computer Scientists [M]. Cambridge: MIT Press, 1991

[16] Scott P J. Some Aspects of Categories in Computer Science [M]. Handbook of Algebra, Vol. 2. Amsterdam: North Holland, 2000: 3-77

[17] Eerke B. Integrating Specifications: Development Relations and Correspondences[C]// INT'02: Integration of Software Specification Techniques, ETAPS 2002 Satellite Workshop. Invited Lecture. http://www.cs.kent.ac.uk/pubs/2002/1360/, 2007-12-01

[18] ISO. Information processing systems-Open systems interconnection-LOTOS-A formal description technique based on the temporal ordering of observational behavior[C]// IS 8807 1989

[19] Spivey J. The Z Notation: A Reference Manual [M] (2nd edition). London: Prentice Hall, 1992

(上接第 66 页)

进行"字符填充";在接收方该构件将接收到的比特流转化为可以被解释的数据帧,并将数据帧头信息发送给构件适配器。成帧构件的另外一个重要功能是将从链路管理构件中获取的信息封装入数据帧。

6.8 链路管理构件

完成比特流发送和接收功能,收集链路信息并传输给成帧构件,管理物理链路并决定公共信道资源的分配方式和各节点的接入方式,使数据帧能通过信道高效地传输。

7 与 TCP/IP 网络体系的融合

构件化网络体系结构主要是针对当前 TCP/IP 网络的传输层和网络层。通过定义新的协议族,构件化网络完全可以运用在现有的网络编程环境下。已有的网络应用程序也只需要少量改写(修改协议族)即可使用在构件化网络环境下。在 MAC 层该协议可以添加新内容,也可以使用现有 MAC 层;如果仅使用现有 MAC 层,构件化协议只需要在资源管理时获取 MAC 层信息。

结束语 本文为解决当前层次化网络体系结构自身的设计缺陷及应用于无线网络中遇到的问题,提出构件化网络体系结构。构件化网络协议设计思想打破层次概念,使网络协议设计与开发更加灵活。新业务和新需求到来时,用户只需加入功能构件,并根据应用要求灵活组织构件排列方式。由于没有层次概念,因此层次概念带来的冗余、各层无交互等问题已不复存在。用户可以将精力全部用于如何调配网络资源以便为应用提供高质量的服务。

参 考 文 献

[1] Daemon J, Rijndael R V. The Advanced Encryption Standard [J]. Dr. Dobb's Journal, 2001

[2] Koblitz N. Elliptic curve cryptosystems [J]. Mathematics of Computation, 1987, 45(177): 203-209

[3] Clark D, Tennenhouse D. Architectural Considerations for a New Generation of Protocols[C]// Proc. of Sigcomm- 90. 1990: 200-208

[4] Tennenhouse D, Wetherall D. Towards an Active Network Architecture[J]. Computer Communication Review, 1996, 26(2)

[5] Lazar A. Programming Telecommunication Network s [C]// Proc. 5th International Workshop on Quality of Service. USA, 1997: 3-24

[6] Boecking S, et al. A Run-Time System for Multimedia Protocols [C]// Fourth International Conference on Computer Communications and Networks(ICCCN'1995). 1995

[7] Siemens A G. Performance and Software Evaluation of the Modular TIP Communication System[C]// 5th Intl. Conf. on Computer Communications and Networks. USA, 1996

[8] Boecking S. Object oriented network protocol [M]. Beijing: Machinery Industrial Press, 2000

[9] Braden B, Faber T, Handley M. From Protocol Stack to Protocol Heap-Role-Based Architecture [C]// First Workshop on Hot Topics in Networking. Oct. 2002

[10] Verikoukis C, et al. Cross-layer optimization for wireless systems: A European research key challenge[J]. IEEE Commun. Mag, 2005, 43(7): 1-3

[11] Djama I, Ahmed T. A cross-layer interworking of DVB-T and WLAN for mobile IPTV service delivery [J]. IEEE Trans. Broadcast, 2007, 53(1): 382-390

[12] Zeng Jia zhi, Xu Jie, Wu Yue, et al. Service Unit based Network Architecture[C]// PDCAT'2003. Proceeding of the fourth International Conference. 2003

[13] Mussabbir Q B, Yao Wenbing, Niu Zeyun, et al. Optimized FMIPv6 Using IEEE 802. 21 MIH Services in Vehicular Networks[J]. IEEE Transactions on Vehicular technology, 2007, 56 (6)

[14] Soliman H. Mobile IPv 6: Mobility in WirelessInternet [M]. Reading, MA: Addison-Wesley, 2004

[15] Zhou B, Marshall A, Wu J, et al. A cross- layer route discovery framework for mobile ad hoc network[J]. EURASIP Journal on Wireless communications and Networking, 2005, 5(5): 645-660