# 摘要

使用软件开发方法的好处应该是增加软件项目的规则性和清晰度。 传统方法是为中大型团队设计的，具有明显分离的静态角色描述以及自上而下的开发过程。 敏捷方法主要针对预算项目规模较小，固定期限，自下而上的开发流程的小型团队。 在本文中，我们为具有多个小型团队的项目提供了一种组合方法，这些项目一起成长为一个具有艰巨期限和低预算的中型项目，称为智能代理组件软件敏捷方法（aMIAC）。 该方法将在案例研究中引入和应用。

## 介绍

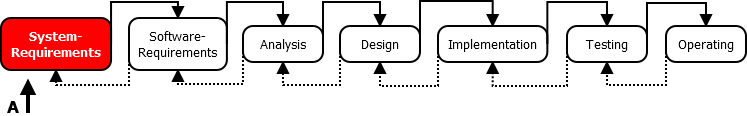
在软件系统的功能不断发展和发展的同时，管理软件开发是计划活动和任务的挑战。过去，工程师试图在工具和框架的支持下有条不紊地计划所有活动。虽然可以安排时间，并且功能具有依赖性，但是软件开发仍是“边做边学”。通过首次开发解决方案或开发人员的未知专业知识来处理未知问题的挑战无异。到目前为止，规划未知项目一直是Sisyphos的挑战。

我们的方法专注于中型项目团队，他们在更改功能请求或开发人员的专业知识方面面临“未知”挑战。整个开发项目几乎没有时间限制。我们试图最小化管理开销，以提高时间消耗和项目沟通效率。最后，我们为每个开发人员提供免费的日间计划，从而最大限度地提高效率。我们在多主体编程范例中组织类似于主体协会的项目团队。

本文的结构如下：在第2节中，我们简要介绍了面向代理的软件工程的开发模型和方法。在第3节中，我们阐述了当前软件开发范例与我们的问题之间的差距。在第4节中，我们提出了管理软件开发的观点，并提出了我们的方法。最后，在第5节中，通过案例研究说明了我们的方法的应用，在第6节中，提出了对我们研究的评估。结论和未来工作，请参见第7节。

## 项目和多项目的标准组织方法

如今，瀑布模型被认为是软件开发过程的标准方法之一。 如图1所示，在瀑布模型中，依次执行不同的步骤。因此，整个开发过程分为多个子任务。



开发人员可以进行下一个任务之前，必须评估和批准当前子任务的结果。如果开发人员无法为当前任务找到解决方案，则他们必须跳回到上一步并重新进行调整。因此，总体目标的实现随着这些子任务的完成而逐步推进。

明确专注于每个完成步骤的评估是一个优势。另一方面，缺点是缺少将项目拆分为子项目并同时开发它们的可能性。该模型的另一个主要问题是灵活性。例如，如图1所示，在实施步骤中几乎无法更改系统或软件需求。在这种情况下，当调整不可避免时，必须停止实施，并且必须重复分析和设计阶段才能继续执行。

一个非常现代的软件工程框架称为Scrum。与瀑布模型相反，Scrum在软件开发过程中基于非常增量和迭代的方法。项目的总体目标分为多个子目标，这些子目标已添加到产品待办事项列表中。产品待办事项列表中的那些项目代表了构建产品必须实现的要求。 Scrum作用基于公司中的经典层次结构。在层次结构的顶部是利益相关者。产品负责人委托产品。他负责产品，并且是利益相关者与产品之间的接口。在产品负责人下面的下一层是Scrum Master，这是一种项目经理，产品负责人与实际开发团队之间的纽带，也称为Scrum团队。根据要实施的项目规模，多个Scrum Master和多个开发团队可以在一个项目上工作。实际的开发发生在所谓的冲刺阶段。那些冲刺有一个确定开发者实现需求的时间。完成冲刺后，将评估当前情况，并可以在产品待办列表进行完善。冲刺又分为“每日Scrum”，在此期间，它简短地讨论了前一天进行的操作，出现的问题以及找到的解决方案。此外，还为这一天应做的事情制定了计划。这种组织形式导致工作并行化和透明化。由于会议频繁，评估的节奏可能很好。无需很长的交接时间和很多开销就可以更改系统或软件要求。这些优势直接来自整体敏捷方法。缺点是层次结构内所需的同步和通信量，以及如何拆分彼此依赖的某些任务的问题。

在开放源代码编程社区中，组织通常结构松散，导致开发人员之间的沟通很少-Xtreme编程模型。由于缺乏组织，每个开发人员都自行选择当前的任务。这可能会导致一个问题，即多个开发人员正在从事同一任务。这种基于自组织和稀疏交流的概念通常被称为“stigmergy-concept”。每个人都选择了似乎是队列中下一个重要任务的任务。这样，小组无需太多协调即可解决总体目标。

这些方法的优点是将重点明确放在解决某些任务上，从而在不增加组织成本的情况下实现整体解决方案。但是，由于开发者之间几乎不存在沟通和协商，因此经济利益损失很大。这种组织模型的主要问题是评估进度和子团队之间缺少同步。后者极大地影响了具有强烈依赖性的子项目的开发。另外，通常仅由开发人员自己对结果进行评估。

## 2.1。面向代理的方法

多重智能代理系统工程（MaSE）主要是一种用于分析，设计和开发异构多主体系统的方法。开发过程分为两个主要阶段。第一个主要阶段是分析阶段，第二个阶段是设计阶段。在分析阶段，目标是根据用户需求确定的，并在目标层次图中进行了组织。之后，定义用例并将其用于创建序列图。在设计阶段，将角色分配给定义的业务代表类，构建业务代表之间的对话，并确定最终的系统结构。

MIAC （智能代理组件方法）专注于旨在重复使用一个已编程的代理和多代理系统的迭代过程。为此，MIAC定义了六个循环安排的开发步骤。在每个周期中，整个系统贯穿于需求管理，系统和用户界面派生，角色建模，实现，集成和部署，请参见图2。此过程通过存储概念进行了扩展，以便可以在长时间运行的多代理系统重复使用代理。

INGENIAS是开发Multiagent Systems（MAS）的方法论和一套工具。作为一种方法，它从五个不同的角度考虑了MAS：组织，代理，目标和任务，相互作用和环境。它提供了一组概念和观点之间的关系，这使开发人员能够描述MAS。 INGENIAS的开发生命周期过程最初采用统一过程。

普罗米修斯是一种开发BDI代理的方法。它包括三个设计阶段：系统规范阶段，详细设计阶段和体系结构设计阶段。系统规范阶段着重于确定代理系统的基本功能。架构设计阶段使用上一阶段的功能来确定系统将包含哪些代理以及它们将如何交互。详细的设计阶段涉及每个代理的内部以及它如何完成与整个系统有关的任务。

GAIA是一种高级的，抽象的，面向代理的分析和设计方法。这个方法主要取决于角色的定义以及将自主的计算实体（代理）视为组织的一部分的含义，就像人类在现实世界中的组织一样。该方法遵循严格的，类似于瀑布的过程模型，用于详细说明高级实体和关系。 Gaia是一种自上而下的全局视图方法，而没有注意到MAS的开发仍是软件工程，其中迭代和增量方法是成功的关键因素。但是Gaia的主要缺点其实际适用性差。盖亚（Gaia）生产非常详细的模型时，绝不会根据要求（或客户反馈）对它们进行检查，也不会例如通过原型进行可行性测试。

## 3.挑战

如前所述，软件工程方法可以分为两类，即静态方法和敏捷方法。敏捷组织模型比静态模型具有更多的迭代和增量方法。最终目标虽然也已确定，但可以在开发过程中进行调整。当处理动态变化的需求时，这显然具有优势，但也可能引起有关子项目的评估和同步的问题。当今的软件已变得越来越复杂。那些功能丰富的项目通常需要以多项目的方式进行处理，以便完全可管理。一个多项目来自一个主要项目，该项目可以分为几个较小的子项目，并由不同的公司或开发人员组同时进行处理。这导致迫切需要一种启用和解决不断变化和不断发展的需求的方法。子项目的分离是分布式系统和项目中成功的关键因素，因为需要子组件的无缝协作才能实现所需的功能。但是，这一重要因素在现有方法中很少得到解决。

Scrum和Xtreme编程模型并不理想，因为它们不能解决前面提到的多代理系统的复杂交互性质。 Scrum中的主要假设是，整个项目团队在相同的工作时间内正在同步工作，这并不成立。 Scrum用于具有一个主要项目的小型团队。 Xtreme编程可以处理分布广泛的团队，但是技术维护人员应指定单个任务。但是，敏捷方法论解决了更改需求的可能性，这是一个重要方面。另一方面，基于系统开发生命周期的方法也不理想，因为它们专注于非常有序的过程和静态项目需求工程。但是评估和保证质量是好的，也是重要的方面。

因此，我们发现，没有一种方法能够完美地解决由中型项目团队或多项目管理人员创建的复杂交互特性的多重智能代理系统。多元代理系统工程方法论确定了最接近的拟合，但是它缺乏在开发过程中轻松更改需求的能力。在此类更改中，开发过程必须从一开始就更改，即分析和设计阶段。

## 4.标准组织方法和敏捷MIAC的组合作为解决方案

如上所述，MIAC方法目前是开发多主体系统的最佳选择，但是在开发过程中调整需求方面缺乏敏捷性。因此，我们建议对MIAC方法进行改编，该方法称为“智能代理组件软件的敏捷方法”，或者简称为aMIAC。在本节中，我们描述了更好地应对动态环境所需的扩展。

由于详尽的目标定义和需求管理对于复杂的系统非常重要，因此在项目开始时我们将使用MIAC的分析和需求管理阶段。与原始方法相反，当获得新的见解时，这种定义和规范可能会在开发过程中发生变化。在确定软件项目的主要目标时，项目团队被分为开发人员小组以应对项目的复杂性。每个小组在角色建模和实现中都只创建一个自治功能-代理。状态，更改和问题在与整个项目团队的周期性会议中处理。为了减少不必要的沟通，应根据项目规模设置会议周期。我们建议使用Scrum方法的交流节奏。

我们建议不要使用Scrum模型的严格层次结构来节省组织成本。相反，组的组织与Xtreme编程的原理相对应。开发人员组应细分为松散耦合的组，并达到一个目标。如果可以自组织，则无需组长。但是，总体进度由产品经理管理。这个角色必须将产品的想法传达给整个项目团队。整个开发小组将采取以下组织步骤，如图2所示。

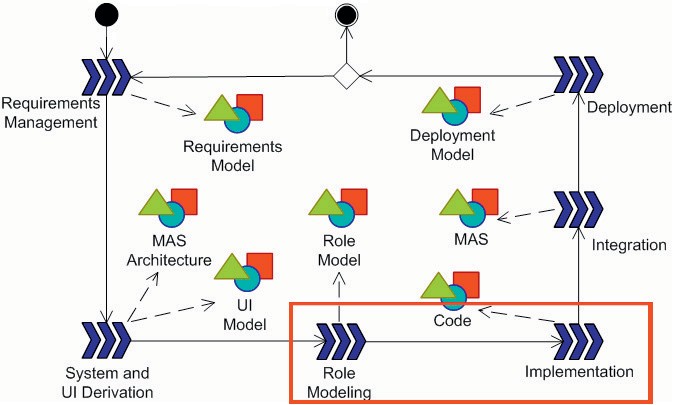


图2.具有六个阶段的MIAC方法论，并行工作的开发人员小组对其进行了改进

在小组级别，不必分配任务，而是每个团队成员都可以选择一个任务来解决所需的功能。该机制类似于Xtreme编程方法，并减少了小型团队中不必要的沟通。根据功能的选择，开发人员将作为本主题的专家介绍给其他小组。这样，问题可以直接与实施人员以及界面中的相应位置进行比较。虽然角色定义和实现是在开发人员组中并行完成的（图2中的红色框），集成和部署将由完整的项目团队完成。不断变化的需求可能需要重新安排现有的开发人员组。

划分为子项目的一个优势是在系统或软件级别中插入更改，如图2中红色框所示。更改仅影响少数开发人员，这些开发人员必须经历MIAC方法的需求管理和系统派生阶段。所有其他开发人员都可以继续他们的工作。

## 5.个案研究

我们在大学的学生课程中运用了我们的方法。该项目时间紧迫，期限紧迫，这是学期结束。必须在3个月的时间内完成并提出所有所做的更改。项目团队（该课程的学生）具有各种兴趣，管理层没有选择他们来实现该项目。该课程由19名学生组成，他们将指导他们达到项目目标。该原型是在JIAC V（基于Java的智能代理组件软件V）框架中构建的。

以智能营养助手为例：总体目标是为周围环境中的用户提供营养方面的咨询。该目标分为五个子目标。每个子目标均由单个代理实现。所有代理在一起形成一个具有人机交互的多代理系统-助手。

多代理系统包含以下代理：内容传递代理能够将食谱的内容提供给系统的其他代理。它通过从互联网检索食谱信息来扩展内容的自治性。库存控制代理可以对家庭中所有可用成分和购物清单进行清点。用户可以添加购买的产品以及数量和有效期。可以列出消费产品的摘要。食谱搜索代理会根据用户的偏好和过敏来自定义用户的搜索。代理会过滤含有可用成分的食谱。如果缺少食谱的成分，则可以将这些项目添加到购物清单中，该清单由库存控制管理。如果选择了食谱，则可以将其传递给Cooking Assistant代理。该代理指导用户准备饭菜。它显示了各个烹饪步骤并强调了重要的说明。烹饪完饭后，将原料从库存控制中丢弃。现代交互组件提供了基于语音和手势的控件，因为在烹饪时，与平板电脑或键盘进行交互并不总是最好的控制方法。

开发过程描述：每个代理商都是由一个学生团体实现的。在课程开始时，通过手势选择该组的成员，换句话说，是根据学生的兴趣领域。没有提出其他建议，如专业知识或个人资料选择。这确保了该方法的独特未知假设是完整的。

每周一次的项目管理会议将介绍代理的当前状态，并通知整个项目团队有关状态和问题的信息。可以与整个项目团队讨论所有问题，以便也包括其他小组的想法。该会议是所有小组的唯一同步点，每周花费两个小时。其他通信通过电子邮件或票务系统处理。如果小组确定了两个特工之间的接口，他们会进行自我协调。

## 6.评估

通过使用我们的方法1，最大的时间消耗是由每周两次的项目管理会议造成的，尤其是考虑到每个开发人员每周的时间预算为13个小时。开发人员平均认为12次会议中有4次是浪费时间。但是，对于大多数开发人员而言，每周的项目管理会议产生了巨大的价值。在对相应开发人员成为严重问题之前，平均已识别并解决了5个非重要问题。但是，与Scrum相比，由19人的开发团队每周召开两个小时的会议，节省了组织成本。

除了每周一次的进度报告会议外，开发人员之间平均大约有20次直接沟通会议，以应对现有问题。因此，已解决了100％的已确定问题，，但不利的一面是，这种直接的交流导致并非所有开发人员都可以获得所有重要信息。由于这种沟通差距，有时会造成混乱并浪费时间。因此，大约有七分之二的情况是由于信息丢失而产生的新问题。为了确保跨组通信而不会丢失任何信息，缺少了一种支持工具，但将来可能会引入它。

通过在项目开始时进行全面的分析和沟通，大多数任务已在项目早期就被确定。大约有三十个中级功能部件，分为五个组。由于子项目，任务分配非常简单。每个开发人员每周从大约12个较小的任务中选择三个任务，这些任务似乎最适合他或她的能力。这使人力资源得以最佳利用，因为任务是由领域内“专家”的积极进取的开发人员解决的。通过这种最佳的能力使用，解决任务所需的时间得以最小化。考虑到这种组织方法的效率的唯一不利方面是由于会议而浪费时间。

最后，该案例研究表明组织模型非常有效。最后，创建了一个复杂的功能多代理系统。但是，必须注意，使用此新模型会出现一些问题。需要改进，特别是在通信领域。另外，整个项目的第一次评估是在整个项目时间表的后期进行的。

## 7.结论与未来工作

本文的目的是为中型项目团队找到一种可以应对未知挑战的合适方法。我们介绍了我们的方法，该方法适用于使用多个项目的多代理系统开发，其中可以通过合理的实施来实现不断变化的需求。为此，已经考虑并评估了不同的方法。由于没有一种现有的方法完全符合需求，因此我们将这些现有方法的要素结合起来以形成新的方法。在案例研究中应用了该新方法，以显示优点并识别薄弱点。

我们的研究和案例研究表明，需要针对多代理系统项目的敏捷软件开发框架。传统的静态方法论涵盖了中型项目，而敏捷的方法则侧重于小型团队规模的项目。现有的组织模型不是最适合在具有动态需求的多代理系统开发项目中使用。一方面，刚性组织模型中的静态需求和几乎不变的需求不适用于具有动态需求的多主体项目。另一方面，敏捷模型不符合多主体多项目的结构和依赖性数量。

因此，我们建议为敏捷多代理项目引入一种新的软件开发方法。在我们的模型中，我们扩展了刚性的MaSE模型，以包括在实施过程中可以不断发展的敏捷目标和需求定义。此外，通讯方式也进行了更改，因此在每周一次的会议上向各小组介绍了总体进展情况。这样，每个开发人员都可以了解进度，并且可以尽早发现问题。

但是，在我们的案例研究中，也发现了一些问题。在某些情况下，需求变更无法通知所有开发人员。这些更改是两次会议之间发现的严重问题的结果。这些问题立即得到解决，并且只有相关的开发人员才知道，因为有关该问题的信息似乎对所有人都不相关。解决问题的状态并非总是被所有开发人员所认可。会议期间发现的问题随后由有关开发人员纠正。但是这种状态变化并不总是传达给整个团队。

为了解决我们组织方法中的这些问题，需要一种工具来减少这些负面影响，甚至完全消除它们。需要使用该工具来增加开发人员之间的协调和沟通。这样，所有成员都可以随时了解实施进度和更改，这需要跨项目的沟通渠道。

更好地概述已发生的问题以及已完成和尚未完成的任务的一种可能性是，使用Redmine2或YouTrack3之类的错误跟踪器应用程序。使用这些工具，所有开发人员都可以了解项目进度。这些工具的问题在于它们不能很好地支持松散和敏捷的层次结构，而是依赖于经理和开发人员的严格层次结构。