软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2021,32(5):1256−1283 [doi: 10.13328/j.cnki.jos.006232]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

面向微服务架构的开发组织适应性评估框架*

崔海涛[1,2], 章 程[1,2], 丁 翔[1,2], 曹伶俐[1,2], 杨 耘[2,3]

[1](多模态认知计算安徽省重点实验室(安徽大学),安徽 合肥 230601)

[2](安徽大学 计算机科学与技术学院,安徽 合肥 230601)

[3](School of Software and Electrical Engineering, Swinburne University of Technology, Melbourne 3122, Australia)

通讯作者: 章程, E-mail: cheng.zhang@ahu.edu.cn

摘 要: 目前,一种称为微服务的架构风格正受到越来越多的关注.它给软件项目带来好处的同时,也影响着使用微服务架构的开发组织.研究目的是明确微服务的使用对开发组织产生了哪些影响,这些影响对于组织来说是优势还是挑战.对此进行了一次系统文献综述,并通过元-民族志对定性数据进行合成,最终得出了使用微服务架构对组织产生影响的 7 个方面,分别是组织结构、自治团队、技术/工具、组织文化、开发人员、DevOps 和通信.同时,基于系统文献综述的结果发现,虽然大量微服务的研究都强调为了充分获取微服务带来的预期收益就必须解决组织问题,但是目前针对组织问题发表的学术文献依然较少,因此将那些可能更接近于工业界观点的、高质量的灰色文献也纳入到工作中.根据系统文献综述的结果以及对定性数据的合成,得出了 4 条更高阶的解释,并提出了一个适应性评估框架.此评估框架可以帮助公司评估并提高开发组织对于微服务架构的适应性,为开发组织在面向微服务开发的过程中提供了指导意见.最后,为了验证所提出的适应性评估框架,面向工业界设计并实施了有针对性的问卷调查和行业访谈,两者的结果验证了所提出的适应性评估框架的有效性.

关键词: 微服务;组织;元-民族志;系统文献综述;评估框架

中图法分类号: TP311

中文引用格式: 崔海涛,章程,丁翔,曹伶俐,杨耘.面向微服务架构的开发组织适应性评估框架.软件学报,2021,32(5):1256−1283. http://www.jos.org.cn/1000-9825/6232.htm

英文引用格式: Cui HT, Zhang C, Ding X, Cao LL, Yang Y. Evaluation framework for development organizations' adaptability to micro-services architecture. Ruan Jian Xue Bao/Journal of Software, 2021,32(5):1256−1283 (in Chinese). http://www.jos.org.cn/1000-9825/6232.htm

Evaluation Framework for Development Organizations' Adaptability to Micro-services Architecture

CUI Hai-Tao[1,2], ZHANG Cheng[1,2], DING Xiang[1,2], CAO Ling-Li[1,2], YANG Yun[2,3]

[1](Anhui Provincial Key Laboratory of Multimodal Cognitive Computation (Anhui University), Hefei 230601, China)

[2](School of Computer Science and Technology, Anhui University, Hefei 230601, China)

[3](School of Software and Electrical Engineering, Swinburne University of Technology, Melbourne 3122, Australia)

Abstract: At present, an architectural style called microservices is receiving more and more attention. While it brings benefits to software projects, it also affects development organizations that use microservices architectures. This study's goal is to clarify the impact of microservices on the organization, whether these effects are beneficial or challenging for the organization. A systematic literature

* 基金项目: 安徽省自然科学基金(2008085MF189, 1908085MF206); 国家自然科学基金(61402007)

Foundation item: Anhui Provincial Natural Science Foundation (2008085MF189, 1908085MF206); National Natural Science Foundation of China (61402007)

本文由"面向持续软件工程的微服务架构技术"专题特约编辑张贺教授、王忠杰教授、陈连平研究员和彭鑫教授推荐.

收稿时间: 2020-09-15; 修改时间: 2020-10-26, 2020-12-15; 采用时间: 2021-01-18; jos 在线出版时间: 2021-02-07

review is conducted and qualitative data are synthesized through meta-ethnography. Finally, seven aspects of using microservices architecture to affect an organization are identified, namely, organizational structure, autonomous team, technology/tool, organizational culture, developer, DevOps, and communication. At the same time, it is also found that although a lot of researches on microservices emphasizes that in order to fully obtain the expected benefits of microservices, it is necessary to solve organizational problems, currently there are few scholarly literatures published on organizational problems. Hence high quality grey literatures, which may be close to the viewpoint of industry, are also included in this study. Based on the result of systematic literature review and synthesis of qualitative data, four higher-order explanations have come up and an evaluation framework is proposed that helps companies evaluate and improve the adaptability of their development organizations to microservices architectures. The proposed evaluation framework provides the guidelines for development organizations adapt to microservices. Finally, based on the framework, this paper designed and conducted an industrial survey and interviews. The results of both confirmed the effectiveness of the adaptability evaluation framework proposed in this paper.

Key words: microservice; organization; meta-ethnography; systematic literature review; evaluation framework

使用单体架构(monolithic architecture)设计的软件系统会随着时间的推移变得庞大且复杂,这使得软件的可维护性和可伸缩性几乎变得不可能[1].当前,宣称有着高可维护性、高可伸缩性和更短发布周期等特点的微服务架构(microservices architecture)受到越来越多的关注和研究.微服务架构将单个应用程序分解为围绕业务功能创建的一组小型服务,这些小型服务可以独立开发和部署,并且这些服务通过轻量级的协议进行通信[2].有一些公司例如 Amazon、Spotify 和 Netflix 等,已经成功地迁移到微服务架构并从中受益.

然而值得注意的是,微服务并不能完美地解决所有由单体架构导致的问题.微服务给软件项目带来好处的同时,也对研究人员和使用微服务架构开发的公司提出了挑战.目前,对微服务产生的挑战的研究大致可以分为技术和组织两个方面,研究人员似乎更倾向于解决技术挑战,如服务发现(service discovery)、服务粒度划分(service granularity division)、安全性(security)等,而聚焦于组织挑战的研究并不广泛,这可能会影响到使用微服务架构的预期收益.在 Conway 的《How do committees invent》一文中,他提出组织的通信结构对其软件架构有直接的影响[3].换句话说:当架构风格发生变化时,使用该架构来开发系统的组织也需要相应地进行变化.这个定律的有效性已经被多次证实,并且同样适用于微服务架构[4,5].因此,本文的目标是明确微服务架构的使用给组织带来了哪些影响,这些影响对组织来说是优势还是挑战,公司如何确定其开发组织是否适应微服务以及如何对组织进行调整以提高对微服务架构的适应性.

基于上述研究目标,本文设计了以下研究问题.

- 研究问题 1:使用微服务架构对开发组织产生了哪些影响?
- 研究问题 2:对于使用微服务架构的公司,如何评估和改进开发组织对于微服务架构的适应性?

本文的贡献是,通过系统文献综述(systematic literature review)得出了使用微服务架构对组织产生影响的 7 个方面:组织结构(organizational structure)、自治团队(autonomous team)、技术/工具(technology/tool)、组织文化(organizational culture)、开发人员(developer)、DevOps 和通信(communication).而且,基于系统文献综述的结果和元-民族志(meta-ethnography)对定性数据(qualitative data)的合成,提出了一个适应性评估框架,以帮助公司判断其开发组织对微服务架构的适应性以及提高适应性.同时,在系统文献综述的执行过程中发现:有关组织方面的研究并不广泛,相关的高质量的中文文献十分匮乏.

最后,为了验证所提出的开发组织适应性评估框架,本文进行了一次问卷调查和一次行业访谈.对 65 份有效问卷的分析结果和与 4 位微服务领域专家的访谈结果,都证明了框架的有效性.

本文第 1 节介绍背景和相关工作.第 2 节为本文所用研究方法以及具体的研究过程介绍.第 3 节对系统文献综述的结果进行讨论并提出开发组织适应性评估框架.第 4 节对框架的有效性进行验证.第 5 节对本研究的有效性威胁进行分析.第 6 节进行全文总结并介绍下一步工作.

1 背景和相关工作

作为一种架构风格,微服务代表着相对较小的、可独立部署的、完成单一目的的自治服务.可以使用不同

的编程语言开发微服务,也可以在最适合其需求的硬件上进行部署[2].并且因为规模有限,所以微服务比单体架构开发的软件有更高的可伸缩性(scalability)、可维护性(maintainability)和容错机制(fault tolerance mechanism).因此,综合多种优点于一体的微服务架构在 2014 年被正式提出后,便获得广泛的关注且关注度逐年上升.但微服务并不能作为所有软件系统开发的可行方案,由微服务带来的众多挑战也有待解决.

Conway 在其名为《How do committees invent》的文章中指出,一个系统的通信结构将直接影响该系统的组织结构.这句话已经在实践中被多次证明是有效的.同时,人们发现微服务架构同样也符合 Conway 法则[1,6].使用单体架构风格开发软件的组织不能直接开发微服务.因此,如果公司想使用微服务架构开发软件项目,则必须对开发组织进行调整,以设计出与微服务架构相适应的开发组织.

Weinreich 等人[5]采访了 10 位微服务设计领域的专家,总结出了使用微服务时相关的设计领域以及领域的重要性排序.他们的结果表明:在 15 个微服务相关的设计领域中,组织设计的重要性排在第 4.他们认为:如何横切团队、明确团队需要处理的任务以及所有团队的集中管理,都是很大的挑战.

Di Francesco 等人[7]对工业中微服务的使用进行了实证研究(empirical research),他们将迁移到微服务过程中遇到的挑战划分为技术挑战(technical challenge)和组织挑战(organizational challenge).与此同时,在对 18 位微服务开发人员的采访中,他们发现:在向微服务迁移的过程中,组织的数量增加了(迁移初期平均 6.7 个,迁移完成平均 8 个),但每个组织的成员数量持续减少(迁移初期平均 8.3 人,迁移完成平均 6.5 人).从这一趋势可以推断:一方面,微服务更倾向于支持小型团队;另一方面,为了适应微服务,组织结构一直在发生变化.

Zhang 等人[8]在一项对 13 家不同类型的微服务公司进行的行业访谈中发现,研究中涉及的公司主要还是依靠经验对组织进行转型.不恰当的组织转型,导致了组织结构与微服务架构的不匹配,这可能会导致开发组织之间需要大量的沟通和开发效率低等问题.正如访谈中的一位参与者所说:“我们知道组织应该跟上架构的相应变化,但我们缺乏如何做到这一点的指南.”这也是本文工作的研究动机之一,本文试图为开发组织在面向微服务开发的过程中提供指导意见.

Halappa 等人[9]也意识到:当架构风格发生改变,组织也需要参与变更以与微服务架构相适应.他们提出了 4 点建议:(1) 暴露出你的团队面临的架构问题;(2) 教育他们使用微服务带来的优势;(3) 对微服务架构进行权衡;(4) 让团队自己做决定.

虽然文章中提出的建议可能有助于组织提高对微服务架构的适应性,但建议所涵盖的点很不全面,且没有任何相关的细节描述.

尽管大多数人都承认:想要获取使用微服务带来的预期优势,除解决技术挑战外,组织的挑战也必须解决[6],但有关组织方面的研究并不广泛.并且,现有的有关微服务开发组织的研究中,也并没有一套全面的、能够帮助评估开发组织对微服务适应性的指南性框架被提出.因此,为了协助使用微服务的公司能够跟上架构的变化,本文专注于组织的角度,围绕微服务对组织产生的影响进行了系统且全面的总结,并提出了一个适应性评估框架以帮助公司判断以及提高开发组织对微服务的适应性.同时希望通过本文的研究工作,激发更多的研究人员对微服务开发组织方面的研究兴趣,改善微服务与开发组织匹配度,从而提高使用微服务的收益.

2 研究方法

本文中所用的研究方法包括系统文献综述、元-民族志、问卷调查以及行业访谈.研究过程分为 3 个步骤.

第 1 步,使用系统文献综述搜集所有与微服务开发组织相关的研究,并明确使用微服务给开发组织带来了哪些影响(回答 RQ1).

第 2 步,基于元-民族志方法对相关研究和影响的描述进行定性数据合成,并结合系统文献综述的结果提出开发组织适应性评估框架(回答 RQ2).

第 3 步,基于框架的内容设计问卷调查和行业访谈,以结合工业界的经验对框架的有效性进行验证.

本文的主要研究过程如图 1 所示.

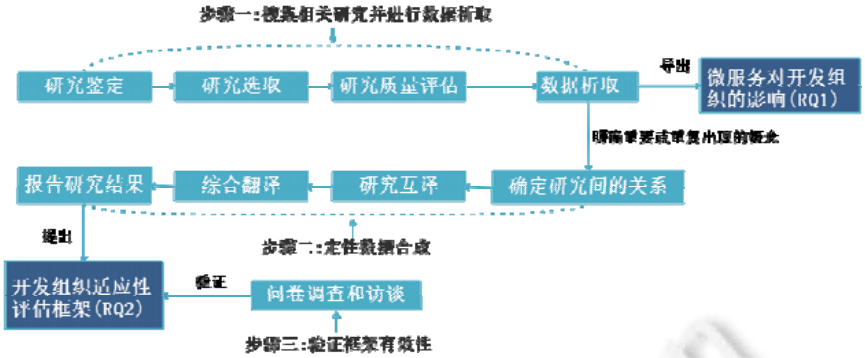


Fig.1 Research process

图 1 研究过程

2.1 步骤一:搜集相关研究并进行数据析取——系统文献综述

本节详细介绍本文所进行系统文献综述的研究过程(包括研究鉴定、研究选取、研究质量评估和数据析取等),这一部分的工作是根据 Kitchenman 等人提出的指导方针[10]进行的,此方针在进行系统文献综述时是经典且常用的.

2.1.1 系统文献综述方法介绍

系统文献综述是以证据为基础的,通过搜索文献,提取数据并进行必要分析的研究框架的使用,系统地、客观地发现并综合一个选定主题的所有可用的经验数据,以回答特定研究问题的研究范式[10,11].该方法的使用可以帮助减少单个初级研究结果中可能出现的偏差.在软件工程中,该方法已经被多次证实有效且取得了大量高水平的研究成果[12,13].当前,在软件工程领域,经典且常用的系统文献综述指导方针(guidelines)由 Kitchenham 等人[10]制定,这份指导方针借鉴了医学领域中的系统文献综述方法.本文其中一个研究目的是系统且全面地了解微服务架构给开发组织带来了哪些影响,并对这些影响给出完整且有效的描述.因此,系统文献综述是适合本文工作的.

2.1.2 索引策略

这一部分用于描述本文中索引文献(即初级研究)的策略.索引策略的目的是识别和收集所有符合本文研究目的的文献.

A) 数据来源

数据来源包括了白色文献和灰色文献.本研究加入灰色文献的目的一方面是增加证据来源;更重要的一方面,是在软件工程中灰色文献可能更接近于工业界的观点,这可以与白色文献的数据互补.

- 白色文献数据来源.主要包括与软件工程相关的电子数据库、会议、期刊:电子数据库包括 Web of Science、IEEE、ACM、Springerlink、百度学术、知网、软件学报、计算机学报等,会议包括 WICSA、ICSA、FSE/ESEC、ASE、EASE、ICSE、ESEM,期刊包括 TOSEM、TSE、IST、ESE、JSS.
- 灰色文献数据来源.广为接受的灰色文献定义是出自第 3 届灰色文献国际会议(ICGL)上提出的卢森堡定义,即灰色文献是在各级政府、学术界、商业界和工业界中产生的,不受商业出版控制的电子版或印刷版的资料.在快速发展的软件工程领域,相关的灰色文献覆盖广泛且发布及时,对领域的发展起到了极大的促进作用.灰色文献既可以作为行业环境中的重要信息来源,也可以在白色文献的研究中作为数据补充[14].因此,本文把高质量的灰色文献纳入到工作中.英文灰色文献来源:Dzone、Stack Overflow 和 Gigaom 等.中文灰色文献来源:CSDN、博客园和 UDN 等.

Google Scholar 可以同时检索到灰色文献和白色文献.

B) 索引串

- 中文索引串

(1) 微服务并且(组织或者团队) 并且 (影响或者挑战或者痛苦或者威胁或者问题或者缺点或者好处或者优势或者优点)

架构更改将不可避免地影响组织的结构,本文试图从采用微服务的企业如何进行企业结构更改的角度来确定微服务架构对组织的影响.所以添加了以下索引串:

(2) 微服务并且 (企业或者公司) 并且 (架构或者体系结构)

- 英文索引串

(1) (microservice OR micro-service OR micro service OR MSA) AND (organization OR organisation OR team) AND (impact* OR influence* OR challenge* OR threat* OR pain* OR issue* OR negative OR drawback* OR disadvantage* OR benefit* OR beneficial OR positive OR advantage*)

(2) (microservice OR micro-service OR micro service OR MSA) AND (enterprise OR company OR corporate) AND (structure OR architecture)

其中,“*”表示模糊搜索.

2.1.3 检索和选取策略

本节介绍文献的检索过程以及文献的纳入和排除标准(inclusion and exclusion criteria).

A) 纳入标准:文献类型为会议文献、期刊、杂志、书籍和高质量的灰色文献.标题和摘要的内容与微服务有关.文章的主要思想是描述使用微服务架构对组织的影响.

B) 排除标准:2014 年之前的文献(2014 年正式提出微服务概念),重复的文章(同一篇论文可能来自不同的数据源,只保留重复文章的一份最新版本).

C) 选择过程:选择过程分为以下 4 个步骤.

(1) 初步搜索分为机器检索(machine search)和手工检索(manual search):机器搜索是在各大电子数据库和学术网站上输入设置的索引词(index terms),自动返回搜索结果;人工检索是作者在相关会议、期刊和技术论坛、博客等寻找符合标准的文献.初步搜索的结果作为最初的文献.

(2) 基于第(1)步获得的最初的文献,采用已经定义的纳入和排除标准进行筛选.当两个作者对是否纳入或是排除一篇文献存在分歧时,第 3 个作者加入讨论以消除分歧,并决定该文献是否保留.

(3) 第(2)步选取的文献由两位作者进行全文阅读及质量评估.符合标准的文献被记录为相关文献;

(4) 本文对相关文献进行滚雪球(snowballing),记录相关文献中有类似工作的参考文献,并重复步骤(3)、步骤(4),以获得与本文有关的最全面的文献.

搜索和选择过程如图 2 所示.

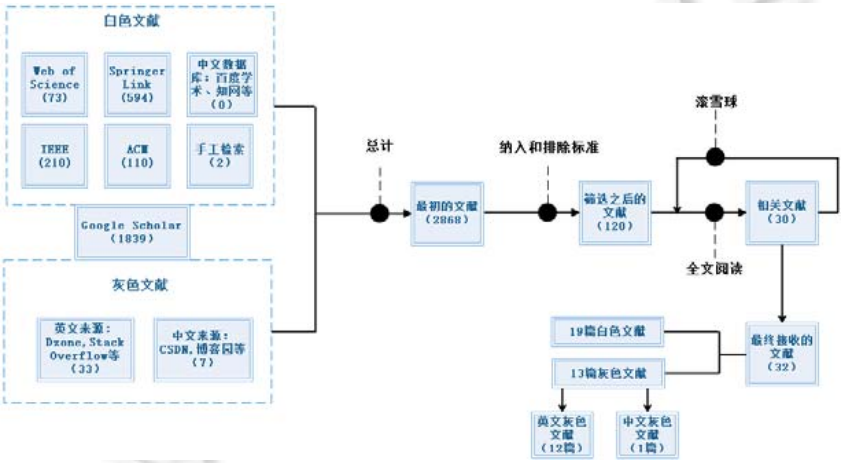


Fig.2 Search and selection process

图 2 搜索和选择过程

2.1.4 质量评估

文献检索完成后,需要通过全文阅读的方式对所选文献的质量进行评估,以获得高质量的论文,保证研究结果的有效性.由于灰色文献不像白色文献有出版社严格定义的格式和同行评审等要求,所以对白色文献的质量评估标准并不适用于灰色文献.因此,本文设计了两套评估标准.尽管评估标准不同,但最终的目的都是获取与本文相关的高质量的文献.质量评估表由两名作者独立完成,当一篇文献的纳入和排除出现分歧时,第 3 个作者加入进行讨论.

白色文献和灰色文献的质量评估表保存在附录 1 中,灰色文献质量评估表参考了 Affan Yasin & Muhammad Ijlal Hasnain[15]的灰色文献评估清单.评估项的取值包括“Y”“P”“N”,其中,“Y”表示是,“P”表示可以部分看出,“N”表示否或不确定(Y=1 分,P=0.5 分,N=0 分).

2.1.5 数据析取

通过对所选文献的反复阅读,我们从文献中析取出能够帮助回答研究问题的数据.数据析取由两个作者独立完成,在对全文进行详细阅读后,填写设计好的数据析取表;然后我们对两位作者析取出的数据进行比较,对不同的区域进行讨论,以确定最合适的析取.当讨论意见没有达成一致时,会请第 3 位作者来解决分歧.一个单独的表格用来记录不一致的析取和每个作者对析取的意见,以便解决分歧和以后的验证.

数据析取的结果将在下一节中展示.

本文通过对 32 篇接收文献(32 篇文献的编号以及信息被保存在附录 2 中,灰色文献已经用“g”标出)的数据析取,得出了使用微服务架构对组织产生影响的 7 个方面,分别为组织结构、自治团队、技术/工具、组织文化、开发人员、DevOps 和通信.这 7 个方面将在第 3.1 节中进行具体讨论.本文使用元-民族志将这 7 个方面定义为常见或重复出现的概念,下一节中将围绕这些概念进行定性数据合成.

2.2 步骤二:定性数据合成——执行元-民族志

本小节对元-民族志方法和执行步骤进行介绍.

数据合成(data synthesis)是系统文献综述过程中不可缺少的一步.数据合成的方法可分为定量分析(quantitative analysis)和定性分析(qualitative analysis).合成数据方法的正确选择和使用决定了系统文献综述结果的可信度[16].在软件工程的初步研究中,经常会包含大量的定性数据[17].对这些定性数据的有效综合,将有利于我们的研究,同时促进该领域的发展.而在本文中就需要对大量的定性数据进行合成,因此本文选择一种名为元-民族志的定性数据方法.选取该方法所进行的考虑如下:

定量数据分析主要应用在实验环境下,适合涉及变量的研究[18].而当我们需要对一个复杂的问题进行深入研究时,定性数据分析是必要的[19].因此在本文中,选择定性数据分析方法是合适且必要的,因为本文的研究目的是去深入了解使用微服务架构给组织带来了哪些影响,并得出更高阶的解释,从而形成最终的适应性评估框架.定性分析方法中还有扎根理论(grounded theory)、主题分析(thematic analysis)和多案例研究设计(multiple case study design)可供本文选择.但扎根理论是从分析单个案例开始,以形成一个理论.本文并不需要形成任何理论,所以该方法不适用.而主题分析强调的是在数据中对主题或模式精准的定位、检查和记录,所以也被排除.本文的研究也并未涉及到不同行业间的案例研究,因此多案例研究设计也不合适.

而元-民族志作为一种众所周知的综合定性数据的方法,首先由 Noblit 和 Hare[20]提出,它利用类比的形式将不同的研究相互转化,帮助研究者综合数据,对定性数据做出更深层次的解释.同时,元-民族志的目的是产出新的解释、模型、概念框架或是理论.该方法的执行过程和最后概念框架的产出都与本文的工作相契合,因此,元-民族志最适用本文的研究.

同时,Fu 等人[16]对系统文献综述和元-民族志的执行步骤进行了映射(两种方法的映射关系以及映射关系介绍被保存在表 1 中),他们发现两种方法有很多相似之处,并且建议可以把确定研究之间的关系(determining how the studies are related)、研究互译(translating the studies into another one)以及综合翻译(synthesizing translations)这 3 个步骤来代替系统文献综述中数据合成的步骤.两种方法的有机结合,避免了进行类似的工作.

Table 1 Mapping relationship between systematic literature review and meta-ethnography

表 1 系统文献综述与元-民族志的映射关系

系统文献综述	元-民族志	映射关系介绍
–	(1) 开始	–
研究鉴定	(2) 决定什么是与最初的兴趣相关	研究鉴定和研究选取映射到决定什么与最初的兴趣相关,因为这3个步骤的目的都是获取与研究相关的文章
研究选取		
研究质量评估	没有相关的映射步骤	–
数据析取	(3) 阅读研究	数据析取映射到阅读研究,这两个步骤都是在选取的文章中找出有用的细节
数据合成	(4) 确定研究之间的关系	数据合成映射到确定研究之间的关系、研究互译以及综合翻译这3个步骤,它们的目的都是对析取出的数据进行合成
	(5) 研究互译	
	(6) 综合翻译	
–	(7) 报告研究结果	–

因为在研究选取(research selection)、研究质量评估(research quality assessment)和数据析取(data extraction)等方面,系统文献综述的过程有更加明确且成熟的指导[10],而元-民族志的定性数据合成方法是最适合本文的,因此在本文中选择使用系统文献综述结合元-民族志的方法,即使用元-民族志中的步骤(4)~步骤(6)代替系统文献综述中数据合成的步骤,并在步骤(7)报告研究结果中提出本文的评估框架.

基于系统文献综述,本文得到了使用微服务架构对开发组织产生影响的 7 个方面,包括组织结构、自治团队、技术/工具、组织文化、开发人员、DevOps 和通信.这 7 个方面被本文使用元-民族志定义为重要或重复出现的概念,接下来介绍基于这 7 个概念执行元-民族志进行定性数据合成的过程.

A) 确定研究之间的关系

这一步的目的是:在不同的研究中寻找常见的或重复出现的隐喻或概念(metaphors and concepts),考虑不同研究中概念之间的关系.Noblit 和 Hare[20]建议把各种研究放在一起,创建一个主题或者隐喻列表,确定不同研究之间如何相关.本文参考了已经发表的使用元-民族志的报告,他们通过网格或者表格的形式保存所有研究的概念和主题[21].本文将使用后一种形式.

本文在 32 篇相关研究中找出了 7 种常见或是重复出现的概念,分别是组织结构、自治团队、技术/工具、组织文化、开发人员、DevOps 和通信.为了更方便地找到这些概念之间的联系,本文设计了一张表格.该表格用于保存包括每项研究中的 7 个概念的相关描述和主要思想/结论(存储在表格的最后一列).根据 Schutz[22]的一阶结构和二阶结构的理论,主要思想/理论,即表格的最后一列被视作二阶解释(关于 N 阶(N=1,2,3)结构的描述见表 2).这一步参考了 Nicky Britten 等人[21]在使用元-民族志进行定性数据分析中的工作.由于表格中包括对 32 篇文献的数据析取和总结,篇幅较大且并不影响对本文的理解,因此本文把完成的表格放在了附录 3 中,对这一步工作感兴趣的读者可以进行查看(提示:表格中的空单元格表示文献的内容没有与该概念相对应的描述.最后一列是与本文的研究相关的每项研究的主要观点/结论.引号内的内容是文献的原文,没有引号的内容是对文献中与本文相关内容的解释).

Table 2 N order structure

表 2 N 阶结构

术语	描述
一阶结构	反映参与者(participant)理解的结构,如所包括的研究报告(通常在文章的结论部分发现)
二阶结构	这些研究的作者对参与者理解的解释(通常可以在文章的讨论和结论部分找到),即文章的主要思想或理论
三阶结构	一阶和二阶结构的综合成为一个新的现象模型或理论

B) 研究互译

这一步是将一个研究中的隐喻或者概念与其他研究中的隐喻和概念进行比较和翻译.元-民族志的翻译方法被分为 3 种类别:相互翻译(reciprocal translations)、反驳综合(refutational synthesis)和论据线综合(lines-of-argument synthesis)[20].

相互翻译是对不同研究的主题进行比较,确保一个关键主题可以从其他研究中捕捉到相似的主题;然后使

用类比的形式,以将一项研究的主题翻译成其他研究中的术语(隐喻).相互翻译需要选取研究的关注点足够相似[23].而在本文中,32 篇相关研究虽然都在讨论使用微服务架构对于组织的影响,但每项研究的关注点有很大差异,例如 S18 中只讨论了自治团队,而 S7 涵盖了所有的 7 个方面,这使得相互翻译很难进行.

反驳综合是当一些研究与其他研究中的解释相互反驳,而对相互反驳的两种解释的关系进行考虑.尽管本文中有一些研究存在着对某方面解释相反的情况,例如,在 S1,S7,S17,S21,S22 中认为微服务给组织通信带来了优势,而在 S2,S8,S10,S25 中认为通信是组织需要解决的挑战,但根据系统文献综述结果的讨论,意见产生分歧的原因在于不同研究的关注点(如 S1 的关注点在团队内部通信,而 S33 的关注点在于不同团队之间的通信)不同,这不能被表述成解释相互反驳.因此,反驳综合也并不适合本文.

而论据线综合是通过综合和解释研究中的一阶和二阶解释来形成新的模式、理解等,这与本文的研究目的匹配.因此,本文选用论据线综合对不同研究进行翻译.

C) 综合翻译

前两步的工作已经得到了每项研究中有关 7 个概念的描述和二阶解释,据此形成了一条论据线,见表 3(前两列).根据论据线,最终得出了 4 种三阶解释,其中,前 3 种三阶解释仍然接近于论据线,第 4 种三阶解释是根据前 3 种三阶解释导出的.

Table 3 Synthesizing translations

表 3 综合翻译

概念	二阶解释	三阶解释
组织结构:组织结构需要改变;组织规模,关系;团队数量增加;架构与组织结构缺乏一致性;组织如何演变	(S1) “探讨有效的微服务文化中的关键因素,包括组织规模、动机、关系以及它们如何影响微服务的发展和目标”	(1) 尽管每种开发范式都强调组织文化、通信和开发人员的技能和经验,但在微服务架构中,这些概念表现出新特点
	(S2) 与单体系统相比,微服务开发需要较少的团队间通信	
	(S3) 使用微服务架构可以增加团队自治性	
	(S4) 与其他架构相比,没有经验的团队成员在开始使用微服务时需要较少的工作	
自治团队:高度自主权;高度责任感;无须等待其他团队做出决定;自治,松散耦合;独立部署;增加团队看不到全局的风险	(S5) DevOps和微服务之间的协作将有助于消除开发和运营团队之间的障碍	
	(S6) 总结了19个在微服务中没有考虑到的因素,并表明,并不是所有类型的公司都能从微服务中获益	
	(S7) 在DevOps和CD环境中,使用微服务可以带来很多优势	
	(S8) 当软件公司足够大时,微服务是处理复杂性和规模的好方法	
技术/工具:异构技术堆栈;技术选择的激增可能很快变得难以控制;避免技术锁定;自主选择工具;重用和故障处理变得困难	(S9) “致力于将微服务与可适应的企业体系结构集成”	(2) 每个概念并非都独立存在,公司需要在概念改变中进行权衡或是制定标准
	(S10) 微服务中的组织,工作方式,设计实践,模块化和任务分配	
	(S11) 提出了9个常见的微服务陷阱及其可能的解决方案	
	(S12) 开发和操作人员的技能被认为是主要障碍	
	(S13) “微服务架构还有很长的路要走,在这个过程中,还存在许多挑战”	
组织文化:文化需要变革;文化更加开放和不受约束;需要组织信任;开发人员对变革的抵触;组织文化评估	(S14) 公司可以在使用微服务中从技术级获益,但需要在组织级进行调整	
	(S15) 探索情景上下文对微服务开发的影响	
	(S16) “调整原则、实践和文化的微服务体系结构”	
	(S17) “根据从业者在开发基于微服务的系统时所经历的不良实践确定了一组(20个)微服务反模式”	(3) 微服务并非万能解,使用微服务之前需要明确使用原因以及公司是否做好充分准备
开发人员:不知道微服务是做什么;缺乏微服务经验和技能;需要高技能的开发人员;团队需要熟悉这些概念的成员	(S18) 一个团队应该只拥有一个服务,这足以确保团队自治和松耦合	
	(S19) 组织结构需要改变,文化需要得到尊重但也必须进行变革	
	(S20) “如何使用微服务管理团队”	
	(S21) “了解部署微服务的主要优势,以及采用基于微服务的产品所必需的公司文化”	
	(S22) 介绍使用单体架构开发应用程序相关的问题以及如何适应微服务架构	

Table 3 Synthesizing translations (Continued)

表 3 综合翻译(续)

概念	二阶解释	三阶解释
DevOps:微服务支持DevOps;需要采用DevOps;结合DevOps元素;DevOps的适应性	(S23) 在微服务架构上使用DevOps的优势,加速了开发和部署等	–
	(S24) 给出了使用微服务架构组织需要如何进行演变的建议	(4) 公司想要获得微服务预期收益必须创建适应该架构的开发组织,公司需要明确影响组织适应微服务的因素和因素之间的关系并合理应对
	(S25) “采用微服务仍然是复杂的,从单一系统迁移成功取决于几个因素”	
通信:团队内部和团队间的沟通;减少团队间的通信需求;缩短客户到负责的开发人员的沟通路径;通信开销被最小化;缺乏沟通;改进沟通渠道;寻找不同的沟通策略	(S26) “对迁移到微服务的背景下的意图、策略和挑战进行了定性研究”	
	(S27) conway定律与组织结构的关系	
	(S28) 微服务不能解决所有问题,同时它与DevOps密切相关	
	(S29) 公司需要在给团队充分自治权与控制之间取得正确的平衡	
	(S30) 团队迁移到微服务的技巧:成功的团队文化,给每个团队成员一个成长的机会和合理的技术选择	
	(S31) 提出了使用微服务时组织会遇到的6个问题	
	(S32) “组织需要走向去中心化、小团队化、全能化和DevOps化”	

本文结合导出的 4 种三阶解释和系统文献综述的结果提出了一套微服务中开发组织适应性评估框架,框架将在下一节中进行详细介绍.

2.3 步骤三:验证框架有效性——问卷调查和行业访谈

基于系统文献综述和元-民族志,本文提出了一套开发组织适应性评估框架,该框架将在第 3 节中详细描述.为了结合微服务从业者的实际开发经验对框架的内容进行讨论和验证,本文开展了一次问卷调查和一次行业访谈.问卷调查和行业访谈基于所提出的开发组织适应性评估框架进行设计,首先要求参与者评估框架第 1 级中的 7 个概念(组织结构、自治团队、技术/工具、组织文化、开发人员、DevOps 和通信)在微服务的组织设计中的必要性;接着,提出框架中对应于 7 个概念进行设计的建议,参与者需要结合微服务的开发经验来评估建议的有效性.问卷调查和访谈的结果证明了框架的有效性.

2.3.1 问卷调查

A) 问卷设计

本文通过两名研究人员进行头脑风暴,列出了所有与框架验证有关的一系列问题和可能的回答,以确保问题是全面的;接着,不断地对问题和回答进行定义和细化以形成问卷初稿,并在一位问卷设计方面有着丰富经验的专家指导下,对问卷进行了 6 次大修;最后,本文邀请了两名志愿者进行了试点测试.

B) 问卷内容

问卷由 4 个部分组成,共包含 21 个问题.问卷见表 4,其中,问题 Q6~Q20 为评估题.本文需要问卷参与者有微服务领域的开发经验,以能够回答设定的调查问题和保证结果的有效性,因此,问卷的第 1 部分要求参与者填写其基本信息(开发经验,主要角色)和所参与项目的基本信息(架构的使用方式、项目的领域、项目的完成周期等).第 2 部分用于提出的验证框架中的 7 个概念在组织设计中的必要性,参与者需要对这 7 个概念进行必要性评级.评级标准参考了李克特量表(Likert scale),等级包括“非常必要”“必要”“一般”“不必要”“非常不必要”.第 3 部分的问题基于参与者第 2 部分的回答生成,对于被参与者认定为“非常必要”“必要”和“一般”的概念给出框架中的建议,并由参与者结合实际开发经验对建议的有效性进行评估,有效性等级包括“非常同意”“同意”“既不同意也不反对”“不同意”“非常不同意”.问卷的最后一部分为一个开放性问题,目的在于了解公司在使用微服务进行开发中,开发组织会遇到哪些问题以及如何对组织进行调整.

C) 识别问卷参与者

为了保证问卷的参与者是有实际微服务开发经验的人员,本文邀请了 4 家国内知名度较高且正在使用微服务架构的公司进行调研.参与调研的公司业务领域广泛,涉及金融、社交、教育、娱乐等.同时,为了收集尽可能多的有效数据,我们邀请这些公司的参与者转发给他们可能认识的有微服务开发经验的潜在参与者.最终,本文共收到了 73 份完整的问卷回复,其中 8 份完成时间在一分钟以内,且连续选择了同一个选项的问卷被识别为无效问卷,因此,共 65 份有效问卷被纳入研究并进行分析.

Table 4 Details of the questionnaire
表 4 问卷的详细信息

参与者基本信息	
Q1. 您从事微服务架构开发多久了?	Q2. 您在公司中担任的主要角色是什么?
项目的基本信息	
Q3. 您的公司是如何使用微服务架构的?	Q4. 您目前开发的微服务项目所属领域是?
Q5. 项目完成需要多长时间?(如果项目仍在进行,请勾选计划完成时间)?	
阶段 1:评估框架中的 7 个概念在实际开发中进行设计的必要性	
自治团队: Q6. 高度自治的团队(团队是松散耦合的,每个微服务团队可以独立开发和交付)	通信: Q7. 良好的沟通结构(团队内部和不同团队之间有效的沟通)
组织结构: Q8. 与微服务相匹配的组织结构(合适的团队数量和规模,完善的人员配置等)	组织文化: Q9. 优秀的组织文化
DevOps: Q10. 与 DevOps 紧密结合(实现持续监控,持续交付等)	开发人员: Q11. 由经验丰富和技能娴熟的开发人员构建团队
技术/工具: Q12. 合理选择开发工具和技术	
阶段 2:评估框架中建议的有效性	
(基于 Q6,Q7) Q13. 建议 1:在开发中加入相关的会议(技术协调会议,依赖协调会议等),以保证项目整体的同步性	(基于 Q7) Q14. 建议 2:在不同团队间加入团队协调员,保证不同团队间信息传递的正确性
(基于 Q8,Q10) Q15. 建议 3:每个微服务的团队规模建议在 6~7 人左右	(基于 Q8,Q10) Q16. 建议 4:每个微服务团队都应该包括开发人员,运维人员和质量保证人员
(基于 Q9) Q17. 建议 5:公司需要考虑团队成员对微服务的接受程度,这种接受体现在团队成员是否愿意放弃原有的工作团队,而有意愿进入新的微服务团队	(基于 Q11) Q18. 建议 6:开发前,对开发人员进行技能和微服务基础知识的培训
(基于 Q11) Q19. 建议 7:确保每个微服务团队中都包含有微服务开发经验的人	(基于 Q12) Q20. 建议 8:微服务团队需要有对技术和工具选择的自主权,但公司必须给出一个可供选择的技术和工具的清单.(目的在于团队可以选择最适合他们的技术和工具,同时又避免技术和工具过量使用导致的可维护性等问题)
开放式问题	
Q21. 在实际工作中,公司会从哪些方面评估开发组织是否适应微服务?以及有哪些具体的措施来调整开发组织?请留下您的宝贵意见,十分感谢!	

2.3.2 行业访谈

A) 访谈设计

为了进一步验证本文所提出的适应性评估框架,同时为了结合工业界的开发经验来细化框架中的建议以帮助公司更直接的采用,本文设计了一次微服务行业内的访谈.访谈的问题提纲围绕框架进行设计,同时由一位在访谈方面有丰富经验的作者进行审阅.通过对问题提纲的 3 次修改之后,最终设计出的访谈主要包括以下 3 个部分.

1) 热身问题:专家的基本信息和所负责项目的基本信息;
2) 探讨框架的内容:我们将适应性评估框架和围绕框架所设计的问题提前发送给参与者,让他们有足够的时间熟悉框架内容并思考问题回复;
3) 实际开发中遇见的组织问题:收集参与者在实际开发中遇见的组织方面的问题以及具体的解决方案.

B) 识别访谈参与者

为了识别有微服务开发经验的参与者保证数据来源的可靠性,本文直接面向国内正在使用微服务架构进行开发的知名公司,并成功邀请到了来自 3 家公司的 4 位微服务领域的专家.

C) 访谈结果

因为地域的原因,本次访谈主要包含 3 次在线访谈和 1 次面对面访谈,并由两位作者分别对访谈内容进行记录;同时,为了避免丢失一些重要的信息,所有的访谈均在得到参与者的许可后进行了录音保存.通过分析访谈记录和录音,本文筛选出了访谈中的主要内容并进行了总结,结果将在第 4 节中进行讨论.

3 主要发现和适应性评估框架

本节将介绍系统文献综述中数据析取的结果(回答 RQ1),同时结合元-民族志对定性数据的合成以及 4 条高阶解释,提出开发组织适应性评估框架(RQ2)并进行讨论.

3.1 微服务对开发组织产生的影响(回答RQ1)

本文通过对 32 篇接收文献(32 篇文献的编号以及信息被保存在附录 2 中,灰色文献已经用“g”标出)的数据析取,得出了使用微服务架构对组织产生影响的 7 个方面,分别为组织结构、自治团队、技术/工具、组织文化、开发人员、DevOps 和通信.7 个方面的证据来源被保存在表 5 中(其中,“*”表示此方面被该研究划分为优势, “#”表示被该研究划分为挑战).这 7 个方面也被作为常见的或重复出现的概念,以进行定性数据分析.有趣的一点是,如图 3 所示,本文发现:文献中对技术/工具、自治团队和通信这 3 个方面的讨论存在差异,它们既被认为是优势,又被认为是挑战.接下来将对这 7 个方面分别进行讨论,并对差异产生的原因做出解释.

Table 5 Source of evidence

表 5 证据来源

编号	组织结构	自治团队	技术/工具	组织文化	开发人员	DevOps	通信
S1	#	–	–	#	–	–	*
S2	–	*	–	–	–	–	#
S3	#	*	*	–	#	–	–
S4	#	–	*	–	#	–	–
S5	#	*	#	–	#	*	–
S6	#	–	*	#	#	*	–
S7	#	*	#	#	#	*	*
S8	#	*	*	–	#	*	#
S9	–	–	#	#	–	*	–
S10	#	–	–	–	–	–	#
S11	#	–	–	–	#	–	–
S12	–	*	*	#	#	*	–
S13	–	#	#	#	–	–	–
S14	–	*	#	–	#	–	–
S15	#	–	–	#	#	–	–
S16	#	*	–	#	–	*	–
S17	–	–	#	–	#	*	*
S18	–	*	–	–	–	–	–
S19	#	–	–	#	–	–	–
S20	#	–	–	–	–	–	–
S21	#	–	#	#	–	*	*
S22	–	*	–	–	–	–	*
S23	#	–	–	#	–	*	–
S24	#	–	–	–	–	–	–
S25	–	*	–	–	#	–	#
S26	–	#	–	#	–	*	–
S27	#	–	–	–	–	*	–
S28	#	*	*	–	–	–	–
S29	–	*	*	–	–	–	–
S30	#	–	*	#	#	–	–
S31	#	–	–	#	–	–	–
S32	#	–	–	–	–	*	–
合计	21	15	15	14	13	13	9

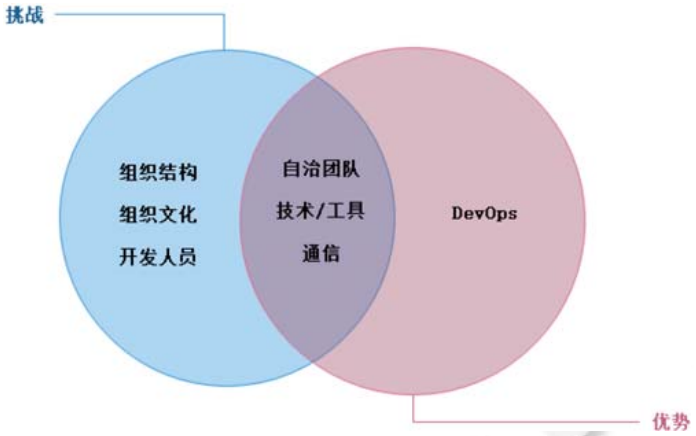


Fig.3 Impact of microservices on organizations

图 3 微服务对组织产生的影响

3.1.1 组织结构

组织结构是当前有关微服务的讨论中关注相对较多的一个话题,调整并改善公司的组织结构,无疑将促进架构的有效性,从而帮助公司获取到使用微服务架构预期的收益[24].人员配置、团队的数量以及每个微服务团队的规模等,都是一个设计良好的组织结构需要去解决的问题.例如,微服务团队中的人员配置发生了变化,开发人员、质量保证人员(quality assurance personnel)和运维人员(operation and maintenance personnel)不再是在各自独立的部门工作,现在每一个微服务的开发团队中,都应该包含这 3 种人员[25].而对于团队数量和团队规模的设计可能不是一蹴而就的,需要在不断地适应微服务架构的过程中,对团队数量和团队规模进行调整.Di Francesco 等人[7]在对微服务的一个行业调查中指出:在完成单体架构到微服务架构的迁移时,每个团队的人数在 2~12 人之间,平均 6.5 人.这是可以进行参考的一个标准.

在团队规模的设计过程中,需要注意的一点是:当公司的项目开发进度已经出现延迟时,向团队中盲目增加人员可能并不能解决问题,并且可能会让项目延迟更加严重[26].新的开发人员加入一个团队进行工作,需要一定的时间去适应,公司需要在评估之后对是否增加人员做出决策.

3.1.2 自治团队

Guzzo 和 Dickson[24]将自治团队定义为“通常执行高度相关或进行相互依存工作的雇员被识别为组织中的一个社会单位,并且他们在许多方面都被赋予了重要的权威和责任,比如制定计划、安排任务,并向成员分配任务和做出具有经济后果的决策等”.

而自治性正是被微服务的支持者所宣称的一种优势[27].团队之间的依赖被尽可能地降低,因此,当某个开发团队完成了自己负责的微服务的开发时,他们不必等待其他团队的开发进度,就可以将完成的微服务发布到生产环境中.同样,当某个团队遇到问题导致进度延迟时,其他团队并不会受到影响.自治性的优势还使开发团队可以更专注于他们的开发,解决遇到的技术难题,而不用在与其他团队的沟通或是互相推卸责任上浪费时间,这无疑提高了整个项目的开发效率[25].

但是有一些研究人员认为,自治团队带来了挑战[28].自治团队可能存在看不到全局的风险,因为他们可以不必经常与其他团队进行协商而独立开发和交付.这种风险可能会导致项目整体的不同步,或是不同团队难以合作的文化.在这种文化中,团队可能需要去修复由其他团队负责的问题.

3.1.3 技术/工具

另一个讨论存在差异的影响是技术和工具的使用.

微服务架构使得每个微服务开发团队拥有了自主权,这种自主权允许开发团队不再局限于使用传统的技术或是受限于整体项目的技术选择,他们可以通过尝试不同的选择以找出最适合开发出所负责的微服务的技术和工具.同时,这也将有利于新技术的发展,并且降低开发人员使用可能已经过时技术的可能性.

但在实际应用中,大量技术和工具的使用将给开发带来挑战,数量会变得难以控制,从而使整个项目变得难以管理[6].特别是在有成员变动的团队,没有代码注释或是技术说明性文档,太多技术的使用可能会降低微服务的可维护性和可扩展性.

因此,公司支持开发团队自由选择技术,同时也需要有一定的控制.开发不同的微服务所需的技术或工具可能并不相同,公司应该根据不同的情况制定不同的标准[29].制定标准时需要注意的是:一味追求新技术或是执着于使用过时的技术,都可能会降低使用微服务架构进行开发的有效性.另外,当一个团队在开发时想要使用一个新的技术堆栈时,必须经过一个评审过程,以确定新技术带来的收益超过引入新的技术堆栈带来的操作开销[30].

3.1.4 组织文化

文化问题如果没有被很好地处理,那么使用微服务所预期的收益可能永远也无法达成[31].Thomas Jardinet也指出:企业迁移向微服务真正的僵局是文化方面,组织和开发人员可能会抵触这种变革,因为他们无法接受微服务[28].例如,在 Fritzsch Jonas 等人[1]的研究中有这样一个案例:一家软件服务公司正在尝试使用微服务架构为汽车客户现代化一个非常大的配置管理系统,这就需要原本的开发团队经历从瀑布式到敏捷式的开发思维转变.这会让一些开发人员难以接受,因为他们意识到之前开发的系统和取得的成就已经过时,而能否在微服务开发中找到自己在团队中的定位,使他们陷入担忧.让团队中的所有人能够接受微服务,这是很困难的事情,也是必须被妥善解决的问题[1].

团队合作中需要异议和讨论,因为在团队进行讨论时很容易被群体思维(群体思维是社会心理学家 Irving L.Janis 提出的一种心理现象)所压倒,即,为了与别人达成共识而放弃表达自己的意见.微服务开发团队需要创造一个让异议和讨论自然而然产生的环境,让团队成员在不同解决方案的讨论中获取最佳的解决方案.组织成员间的信任也是组织文化的组成部分,需要相信其他开发人员能够开发出可靠有效的微服务[31].

公司可以通过 Westrum 文化模型(Westrum cultural model)的使用来对团队文化是否健康进行评估,Westrum 文化模型如图 4 所示.Westrum 文化模型不是专门为微服务设计的,但使用微服务架构的公司可以使用这种模型来了解公司的文化状态,评估组织文化是否健康.Westrum 文化模型把公司的文化氛围分为 3 种:病态型(pathological)、官僚型(bureaucratic)和生成型(generative).病态型的文化特征是团队中存在着大量威胁和恐惧,团队间的低效合作和消极文化可能会造成人员流失.官僚型的文化中,人们倾向于维持等级制度,团队中充斥着规则,成员很少关心组织的总任务.生成型的文化中成员间具有高度的合作和信任,他们关注的重点是团队的任务,而不是自己[32].

Fig.4 Westrum culture model

图 4 Westrum 文化模型

3.1.5 开发人员

正如 Elger Peter 等人[33]所说:跨功能实施团队的技能,将在很大程度上决定微服务架构的有效性.当开发人员缺乏使用微服务的开发经验,或者甚至不了解微服务架构的基本概念时,他们仍会使用之前开发单体架构的思想去迎接微服务.由这样的开发人员组建的团队将十分危险,因为分布式系统开发(distributed system development)的大泥潭将让开发人员寸步难行.同时,微服务还要求开发人员对系统容错、安全性和延迟等非功

能性方面有深入的了解,这就对开发人员提出了更高的要求.

因此,有经验的开发团队可能更容易获取使用微服务开发带来的效益,而缺乏技能和经验的团队更建议使用传统的单体架构进行开发[4].在将一个商用系统使用微服务架构进行重构的过程中,Armin Balalaie 等人[34]发现,公司需要向新手开发员讲解有关微服务的概念和技术,这浪费了大量的时间但效果并不理想.因此他们认为,公司需要由技能熟练且具有开发微服务经验的人来组建团队.

解决方案:当时间充裕时,对那些新手开发人员进行培养,让他们对微服务的概念有一定的了解以及掌握那些为了完成开发所需的工具和库[34];而当时间有限时,由那些在微服务的开发中有着丰富经验且技能娴熟的开发人员组建开发团队将更有成效;微服务宣传的技术异构性具有很强的迷惑性,多种技术和工具可供选择似乎是一个不错的优势,但是掌握这些技术将给开发人员带来很大的压力,公司应该对不同的微服务开发做出不同的技术选择标准.

3.1.6 DevOps

DevOps 由 Development(开发)和 Operations(运维)组合而成,它是一组实践,其目的是打破开发人员和运维人员之间的壁垒,在保证软件质量的前提下,尽可能减少修改系统和将系统修改部署到生产环境中的时间.持续交付(continuous delivery)和持续监控(continuous monitoring)等任何能实现上述目标的技术都被认为是 DevOps 实践[34,35].

微服务架构与 DevOps 是相辅相成的.尽管在单体架构中,DevOps 也可以进行应用,但微服务所提倡的小型团队,使得 DevOps 的实施更加有效.同时,每个微服务可能都拥有一个单独的后端,这些单独的后端由不同的团队使用不同的技术开发,如果没有 DevOps,由此产生的复杂性会变得难以管理.

3.1.7 通 信

通信可以分为同步通信(会议、视频、电话等)和异步通信(邮件、短信等)两种,团队中良好的通信可以提高软件过程的质量[36,37].当团队中有 M 个人时,需要建立 $M(M-1)/2$ 条通信路径,例如 2 个人仅需建立 1 条、5 个人建立 10 条、10 个人则需要建立 45 条(如图 5 所示).这代表团队成员越多,通信开销和管理开销就越多.正如 Richard Hackman 所说,大型团队通常只会浪费每个人的时间[38].而微服务所提倡的小团队恰恰降低了通信开销,使团队更容易保持凝聚力.

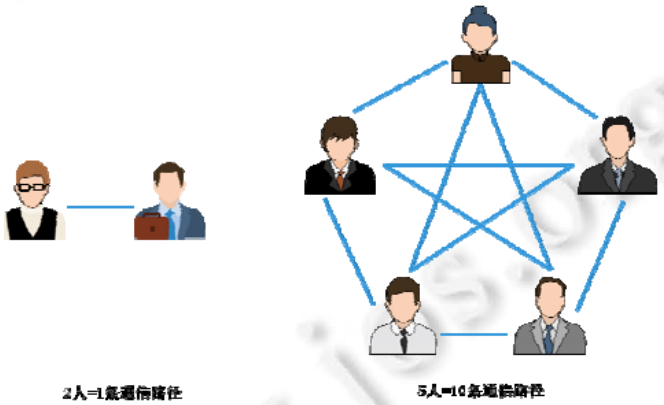


Fig.5 M personal communication paths

图 5 M 个人时的通信路径

Chen[30]在为博彩公司 Paddy Power 使用微服务架构实现 DevOps 和持续交付时发现:客户现在可以直接与负责相关微服务的团队中的人进行通信,而不需要再找微服务系统的负责人.这有助于开发人员做出更快的响应,从而缩短开发周期.因此,一些研究人员认为,微服务对组织的通信带来了优势.

然而,有些研究人员认为,使用微服务架构给通信带来了挑战[13,39].文献中关于微服务中通信问题的讨论存在差异的原因在于侧重点不同:在一个独立的微服务开发团队内部,开发人员之间的通信开销减少;而在不同的

微服务团队之间,由于微服务允许团队独立工作,团队之间的通信需求减少,这在短期可能会带来收益,但会在项目的同步性问题上带来问题[39].因此,为了保证项目的整体同步性,可能需要在开发时在团队中加入技术协调会议(technical coordination meeting)、依赖协调会议(dependency coordination meeting)和执行概要会议(executive summary meeting)等[9].

通信问题需要注意的点是:需要根据团队结构识别通信场景,以制定通信策略.Lenarduzzi 等人[39]把通信分为以下 3 种场景:并置团队(所有的团队在同一地点工作)、有重叠工作时间的分布式开发团队(不同的微服务团队在不同的地址位置工作,但在工作日有一定的时间进行交流)和无重叠时间的分布式开发团队(在工作日无交流),可以通过引入层次结构(即团队协调员)来解决通信问题(具体的通信策略可以参考 Valentina 等人[39]的工作).

3.2 开发组织适应性评估框架(回答RQ2)

这一部分将报告执行元-民族志的结果并进行讨论,即元-民族志的第 7 步(报告研究结果).基于微服务给组织带来影响的系统文献综述的结果,本文从 32 项研究中找出了 7 个重复出现或是常见的概念:组织结构、自治团队、技术/工具、组织文化、开发人员、DevOps 和通信.通过分析这 7 个概念的描述以及 32 项研究的主要思想或理论,形成了一条论述线,并最终导出了 4 条三阶解释.分别为:(1) 尽管每种开发范式都强调组织文化、通信和开发人员的技能和经验,但在微服务架构中,这些概念表现出新特点;(2) 每个概念并非都独立存在,公司需要在概念改变中进行权衡或是制定标准;(3) 微服务并非万能解,使用微服务之前,需要明确使用原因以及公司是否做好充分准备;(4) 公司想要获得微服务预期收益,必须创建适应该架构的开发组织,公司需要明确影响组织适应微服务的因素和因素之间的关系并合理应对.其中,前 3 条三阶解释既与原始结果一致,又超越了原始结果.第 4 条三阶解释是基于前 3 条三阶解释提出的,同时是为了回答本文所设置的研究问题二,即对于使用微服务架构的公司,如何评估和改进组织对于微服务架构的适应性?本文将提出一个适应性评估框架,以帮助公司判断其开发组织对于微服务架构的适应性,并讨论如何提高其适应性.接下来,首先对前 3 条三阶解释进行分析和讨论,以帮助更好地形成框架.

(1) 尽管每种开发范式都强调组织文化、通信和开发人员的技能和经验,但在微服务架构中,这些概念表现出新特点.

无论在敏捷过程还是极限编程等不同的软件过程中,还是使用单体架构开发软件时去创建团队,形成优秀的组织文化、构建高度的通信以及寻找技能娴熟和经验丰富的开发人员都是被提倡的,所以对组织文化、通信和开发人员的技能和经验的考虑并非是微服务专属.但我们必须明确这 3 个概念在微服务中表现出了哪些新特点,这对组织适应微服务架构不可或缺.

微服务架构中的组织文化要求团队成员接受微服务,即开发人员现在需要打破使用单体架构进行开发时的固定思维,了解微服务开发特性,从而明确自己在新的微服务开发中的角色;微服务中通信问题的特点在于客户可以直接与相关微服务的开发人员沟通,这更有利于问题的解决,然而,分布式的微服务开发要求不同类型团队找出与团队组成形式匹配的通信方式;在开发人员的技能和经验方面,虽然 Jonas Fritzsch 等人[40]采访了 10 家采用微服务的公司的软件专家后发现,Restful HTTP、Docker container 和 Java 仍然是开发最常用的工具或技术,但由于微服务技术异构性的特点,可能随时要求有新技术或工具的加入,这对开发人员提出了更高的要求.

因此在微服务中,对于组织文化、通信和开发人员的技能和经验的讨论是必需且独特的.

(2) 每个概念并非都独立存在,公司需要在概念改变中进行权衡或是制定标准.

本文通过系统文献综述找出了使用微服务架构对组织产生影响的 7 个概念,需要注意的是:就像一家公司是一个有机的整体,这些不同概念也是一个整体,并非独立存在.公司需要在不同的概念改变中进行权衡或是制定标准.例如,由于微服务给了团队高度的自治权,不同团队可以不必依赖于其他团队而独立地开发和交付,这使每个微服务团队可以更专注于他们的开发任务.但团队之间的这种松散耦合导致了不同团队之间通信需求的减少,缺乏通信可能会造成整个项目的不同步等问题.又比如,微服务开发需要与 DevOps 紧密结合,这给微服务团队带来了持续交付和持续部署等能力,但是 DevOps 需要打破开发人员和运维人员之间的壁垒,运维人员、

开发人员和质量保证人员在微服务中不再是在独立的部门工作,而是每一个微服务团队的人员配置都需要包含这 3 种人员.又或是需要将一个新的开发人员加入到一个微服务团队时,不仅需要考虑这个开发人员拥有的技能和开发经验,还要对组织结构中的团队规模是否合理进行评估.同样,在微服务开发中,引入一个新技术时,公司需要对引入新技术带来的收益和培养开发人员的支出进行收益评估.

(3) 微服务并非万能解,使用微服务之前,需要明确使用原因以及公司是否必须迁移向微服务.

微服务并不是解决架构问题的万能解,微服务也并不适合所有的公司,大公司更有可能从微服务中获益,小公司盲目跟随这一趋势可能弊大于利.公司在迁移向微服务架构之前,需要明确迁移的原因以及公司是否做好充分准备.通过对迁移向微服务架构的文献总结,我们列出了文献中常见的迁移原因.

① 项目规模的不断扩大,传统的单体架构无法胜任[1];

② 被微服务所宣称的优势(可扩展性、可维护性、独立开发和独立部署等)吸引;

③ 被其他公司成功迁移向微服务的案例所吸引[41].

如果公司尚未开始使用微服务架构进行开发,请先明确迁移向微服务架构的原因以判断是否必须迁移向微服务.如果需要进行迁移,可以根据适应性框架组建开发组织.

如果公司已经迁移向微服务架构,适应性评估框架可以帮助判断公司的开发组织对于微服务架构的适应性,同时,框架中给出的建议可以帮助提高开发组织的适应性.适应性框架如图 6 所示.

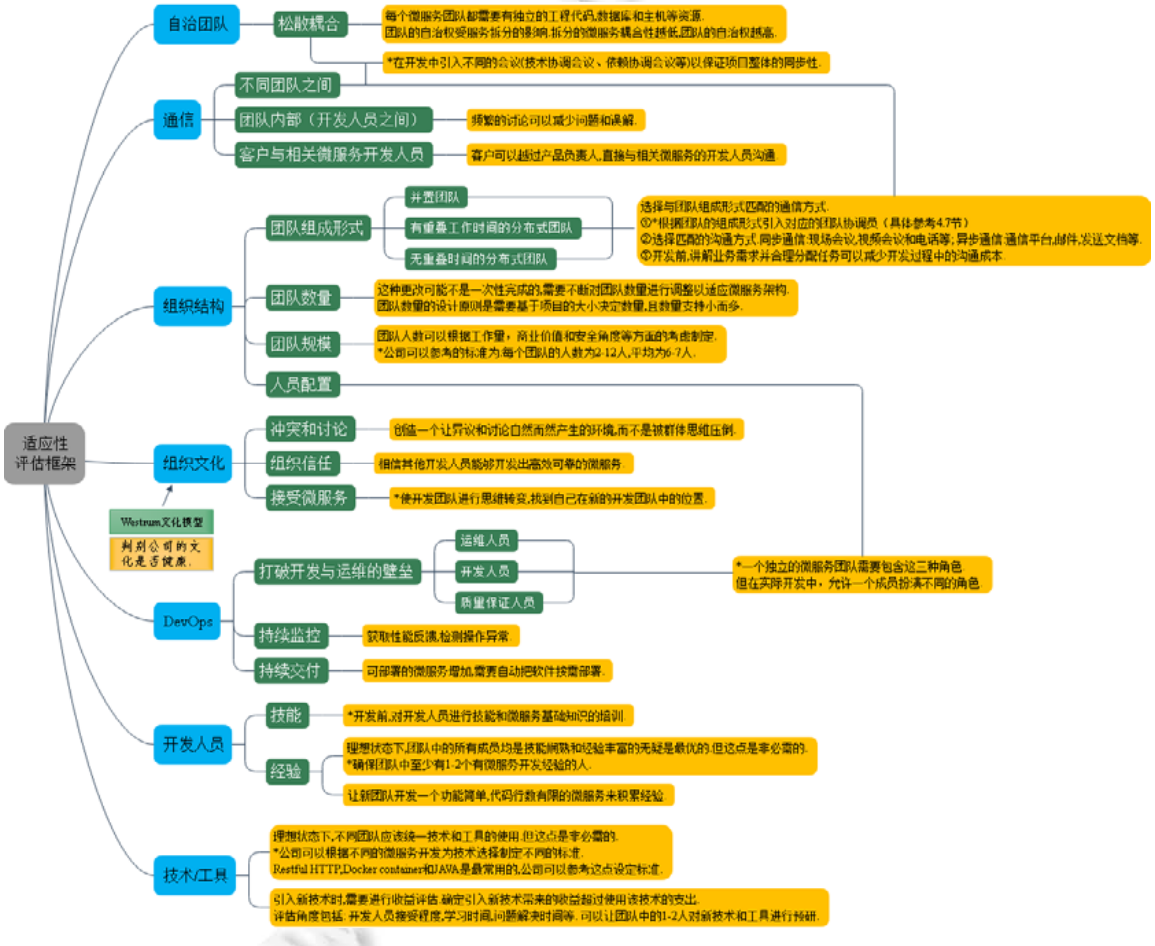


Fig.6 Assessment framework for adaptability

图 6 适应性评估框架

框架的第 1 级为系统文献综述总结出的 7 个概念,包括组织结构、自治团队、技术/工具、组织文化、开发人员、DevOps 和通信.每个概念又产生多条分支,即框架中的第 2 级,公司可以通过判断这些分支(即绿色方框中的内容)是否进行考虑和有效应对,来评估组织对于微服务架构的适应性.框架的第 3 级(黄色框中的内容)是对于需要考虑的因素的建议或者详细描述,目的是帮助公司有效应对,以提高组织对微服务架构的适应性.框架中由文献导出的建议已经用"*"标记出,这些建议将由问卷的参与者结合实际开发经验来评估有效性.

4 验证开发组织适应性评估框架

为了对所提出的框架进行验证,本文基于框架设计了一次问卷调查和一次行业访谈.通过对 65 份有效问卷的结果和与 4 位微服务领域专家的访谈结果进行分析,框架的有效性得到了证明.

4.1 参与者背景

问卷参与者背景如图 7 所示.由于微服务 2014 年才被正式提出,而国内在微服务领域仍处在尝试和探索阶段,因此微服务从业人员的开发经验有限.如图 7(a)所示,本次问卷的参与者中微服务开发经验在 2 年以下的有 36 人(55.4%),2 年以上的有 29 人(44.6%).其中,角色占比最大的为开发人员(41/65),其次为测试人员(11/65),同时也包括架构师、项目经理和高级经理等角色.项目背景如图 8 所示,包括微服务的使用方式、项目所属领域和完成周期.其中,大约一半的参与者开发的微服务项目属于金融领域.项目完成时间或计划完成时间的统计结果呈现多样化分布,时间少至 2 个月以内,多至 6 年以上.

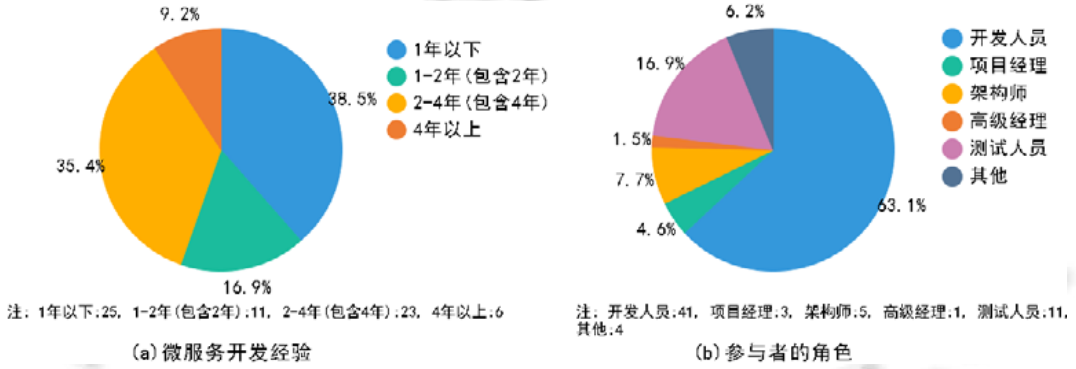


Fig.7 Basic information about the participants

图 7 参与者基本信息

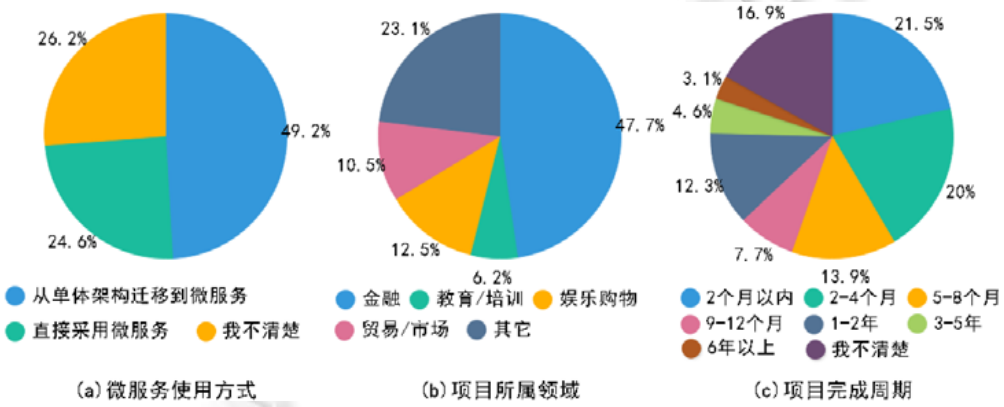


Fig.8 Basic information about the projects

图 8 项目基本信息

本文直接面向国内知名且正在使用微服务架构的公司进行访谈,成功邀请到了来自 3 家公司的 4 位微服务领域的专家,他们均在微服务的设计或是开发中扮演着重要的角色,其中包括高级开发工程师(3/4)和项目经理(1/4).参与者均有着 3.5~6 年以上的开发经验,其中 2 位参与者有 5 年多的微服务开发经验,另外 2 位的微服务开发经验也在 3 年左右.同时,参与者所在的公司涉及业务广泛,主要包括通信、安全、金融等领域.

4.2 验证结果和讨论

本节将结合问卷和访谈的结果,分别从组织结构、自治团队、技术/工具、组织文化、开发人员、DevOps 和通信这 7 个概念及其对应建议(由文献导出的建议,框架中使用“*”进行标记)来展示验证结果,并分别进行讨论.7 个概念和建议的对应关系如问卷表 4 所示.问卷评估结果如图 9 和图 10 所示.

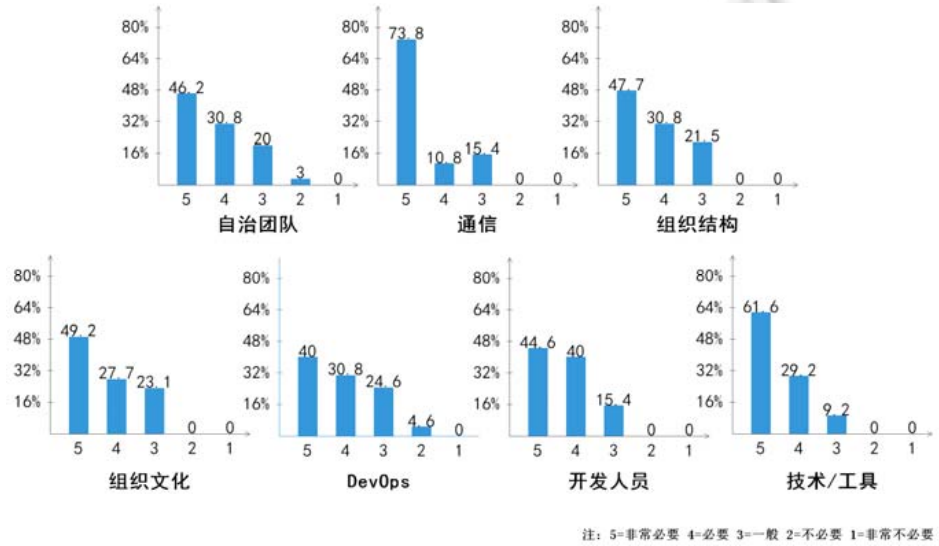


Fig.9 Assess the effectiveness of the seven concepts

图 9 7 个概念有效性的评估

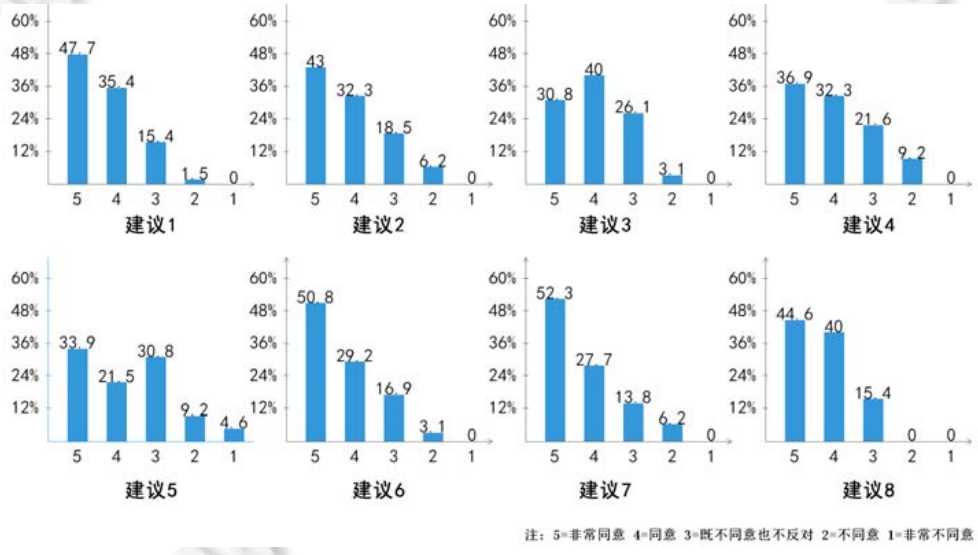


Fig.10 Assess the effectiveness of the suggestions

图 10 建议有效性的评估

A) 自治团队:高度自治的团队(团队是松散耦合的,每个微服务团队可以独立开发和交付)

对应建议——建议 1:在开发中加入相关的会议(如技术协调会议、依赖协调会议等),以保证项目整体的同步性.

1) 赞成:77%的问卷参与者认为,微服务中自治团队的设计是“必要”或“非常必要”的.同时,4 位专家都认为团队需要自治权,并且自治权体现在每个微服务团队都需要有独立的工程代码、数据库和主机等资源.但其中一个专家提到:“微服务团队的自治权依赖于服务拆分,服务拆分不合理可能导致不同团队之间有业务交叉,团队可能需要依赖其他团队的数据或资源,因此会影响到团队的自治权”.

团队的自治权使得不同团队之间的通信需求减少,过少的团队间的通信可能会在项目的整体同步性上带来问题,因此框架中建议在开发中加入相关的会议,以保证项目整体的同步性.超过半数以上的问卷参与者“同意”或“非常同意”这个建议.

2) 不同的意见:仅有 3%的问卷参与者认为自治团队的设计是“不必要”的.而对于建议 1,有 15.4%的问卷参与者认为这条建议的重要性“一般”,且仅有 1.5%的参与者持“不同意”的态度.

3) 结论:构建自治团队的必要性及其建议有效性得到了证明.

B) 通信:良好的沟通结构(团队内部和不同团队之间有效的沟通)

对应建议——建议 2:在不同团队间加入团队协调员,保证不同团队间信息传递的正确性.

1) 赞成:73.8%的问卷参与者认为,微服务中通信结构的设计“非常必要”.4 位专家也认为,微服务的通信结构设计是至关重要的.通信结构主要从同步通信和异步通信两个角度进行设计.一位专家指出:在实际开发中,同步通信方式包括站立会议、项目周会、线上会议等;异步通信可以借助一些通信平台,可以支持开发人员在平台上抛出问题、汇报进度和提出需求等.同时,在正式开发前,公司向开发人员讲解业务需求并合理分配任务,可以减少在开发过程中因为任务不清等问题造成的沟通成本.

文献中认为,公司需要在不同团队之间加入团队协调员.75.3%的问卷参与者同意该建议.2 位专家同意该建议,但他们也表示在目前的微服务开发中,并没有专门设置团队协调员的角色.当不同团队间出现问题时,一般由项目负责人或团队负责人进行解决.

2) 不同的意见:除 15.4%的问卷参与者认为通信结构的设计必要性“一般”外,没有问卷参与者认为通信结构设计是不必要的.针对建议 2,6.2%的问卷参与者不同意这条建议.一位专家指出:在开发中并未发现团队间出现问题,因此团队协调员是冗余的.

3) 结论:构建微服务开发组织中通信结构的必要性及其建议有效性得到了证明.

C) 组织结构:与微服务相匹配的组织结构(合适的团队数量和规模,完善的人员配置等)

对应建议——建议 3:每个微服务的团队人数建议在 6~7 人左右.

1) 赞成:78.5%的问卷参与者认为,微服务的实际开发中组织结构的设计是“必要”或“非常必要”的,并且所有的专家均表示,良好的组织结构有利于提高微服务所带来的收益.但是,适用于所有公司的团队规模设置并不存在,公司需要根据具体的项目规模以及划分的微服务数量来决定团队规模.团队人数的设置可以根据项目工作量、商业价值和安全角度等方面的评估进行制定.

70.8%的问卷参与者“同意”或“非常同意”每个微服务团队人数标准为 6~7 人.同时,3 位专家认为,每个团队的人数应该在 10 人以内,而 6~7 人是实际开发中微服务团队可以参考的人数标准.

2) 不同的意见:对于组织结构的设计,并没有问卷参与者选择了“不必要”或“非常不必要”.同时,仅有 3.1%的问卷受访者“不同意”建议 3,1 位专家对该建议保持中立.

3) 结论:组织结构设计的必要性及其建议有效性得到了证明.

D) 组织文化:构建优秀的组织文化

对应建议——建议 5:公司需要考虑团队成员对微服务的接受程度,这种接受体现在团队成员是否愿意放弃原有的工作团队,而有意愿进入新的微服务团队.

1) 赞成:形成良好的组织文化,有助于团队的发展和项目的开发交付.接近 50%的问卷参与者选择了“非常

必要”,27.7%的问卷参与者选择了“必要”.所有的专家认为:无论使用单体架构或微服务架构进行开发,组织文化的设计都是至关重要的.一位专家指出:“团队成员间频繁的沟通、积极地分享工作和问题以及定期举办活动,都有助于形成良好的组织文化”.

微服务中,组织文化有一个特点,即可能存在部分开发人员不愿意离开单体开发团队,因此,公司需要考虑团队成员进入微服务团队的意愿.55.4%的问卷参与者同意该建议,且一位专家指出:其公司会根据开发人员的经验、工作意愿以及繁忙度等因素来挑选人员构建团队.

2) 不同的意见:没有任何一个问卷参与者认为构建良好的组织文化是“不必要的”或“非常不必要的”.建议5与实际开发中的实践存在一定的差异,13.8%的问卷参与者不同意该建议,且3位专家对该建议保持中立,在他们所进行的微服务开发中,并未出现开发人员抗拒进入微服务团队的案例.

3) 结论:微服务中组织文化设计的必要性得到了证明,建议5的有效性得到了的证明但与实际的微服务实践存在一定的差异.

E) DevOps:与DevOps紧密结合(实现持续监控,持续交付等)

对应建议——建议4:每个微服务团队都应该包括开发人员、运维人员和质量保证人员.

1) 赞成:问卷结果表明,70.8%的问卷参与者认为微服务中需要DevOps,4位专家也均认为微服务中DevOps的应用是必不可少的.

DevOps的一个特性是打破开发和运维之间的壁垒,因此使用微服务进行开发,框架中建议每个微服务团队都应该包括开发人员、运维人员和质量保证人员.本条建议在问卷参与者中支持率为69.2%.4位专家认为本建议可行,他们指出,开发人员、运维人员和质量保证人员在微服务团队中是肯定需要的.但在实际开发中,可能一人会扮演多个角色.例如,开发人员和运维人员可能同时也是质量保证人员.

2) 不同的意见:有少数的问卷参与者(4.6%)认为,微服务并不需要DevOps.同时,有9.2%的问卷参与者不同意建议4.

3) 结论:DevOps的必要性及其建议的有效性得到了证明.

F) 开发人员:由经验丰富和技能娴熟的开发人员构建团队

对应建议——建议6:开发前,对开发人员进行技能和微服务基础知识的培训;

对应建议——建议7:确保每个微服务团队中都包含有微服务开发经验的人.

1) 赞成:除15.4%的问卷参与者选择了必要性“一般”的选项,剩余的所有问卷参与者(84.6%)均认为,团队需要由经验丰富和技能娴熟的人员进行构建.4位微服务领域的专家表达了支持态度,但他们也强调,团队中所有成员均是技能娴熟且经验丰富的当然是最优解.但在实际开发中,这点并非是必需的或者很难实现.团队中一定会存在新手开发人员或未接触过微服务开发的人员,公司在开发前对这些人员进行培训,同时确保每个团队中有1~2个有微服务开发经验的人即可,这也证明了框架中建议6和建议7的有效性.

团队中不可避免地会加入新手开发人员或没有微服务开发经验的人员,因此框架中建议对团队中的人员进行技能和微服务基础知识的培训,以避免在开发中陷入一些常见的技术问题.同时需要确保每个微服务团队都有微服务开发经验的人,以在开发过程中帮助团队成员解决遇见的问题.各有80%的问卷参与者同意建议6和建议7.

2) 不同的意见:仅3.1%的问卷参与者不同意建议6,6.2%的问卷参与者不同意建议7.

3) 结论:微服务中开发人员的必要性及其建议的有效性得到了证明.

G) 技术/工具:微服务团队合理选择开发工具和技术

对应建议——建议8:微服务团队需要有对技术和工具选择的自主权,但公司必须给出一个可供选择的技术和工具的清单.

1) 赞成:90.8%的问卷参与者认为:在实际开发中,每个微服务团队对开发工具和技术的合理选择是“必要”或“非常必要的”.且4位专家表示:每个团队合理地选择开发工具和技术,可以防止技术和工具的过量使用导致的项目可维护性差等问题.

建议 8 的目的在于确保团队可以选择适合他们所负责的微服务的技术和工具,同时又避免技术和工具过量使用导致的可维护性等问题.除 15.4%的问卷参与者未表达观点外,其余的问卷参与者(84.6%)均认为,这条建议适用于实际开发中.2 位专家认为本条建议可行.

2) 不同的意见:有 2 位专家认为:在实际开发中,不同团队统一技术和工具的使用是最理想的,这将有利于减少由于技术和工具选用不一致导致的问题而消耗的时间成本.

3) 结论:技术/工具的必要性及其建议的有效性得到了证明.

本文首先从框架中的第 1 级,即 7 个概念在微服务的组织设计中的必要性进行评估;接着导出由框架中总结出的、每个概念设计时的建议,并由问卷和访谈的参与者,结合实际微服务开发经验来评估这些建议的有效性.通过问卷和访谈的结果进行分析和讨论,证明了本文所提出的开发组织适应性框架的有效性.同时,本文结合访谈中专家的实践经验,对框架中的建议给出了更具体的实现细节,这有助于使用微服务的公司更直接地采用框架.

5 有效性威胁

尽管本文中系统文献综述和元-民族志综合定性数据的执行过程都严格按照指导方针进行,但有效性的威胁(threats to validity)依然存在,这一节将对可能出现的有效性威胁进行讨论.

5.1 相关文献

本文中系统文献综述部分的工作根据 Kitchenham 等人提出的指导方针,严格制定了数据检索过程.具体的检索被分成了机器检索和人工检索,同时,选取的数据来源(中英文电子数据库、会议和期刊等)足够全面.但即使如此,可能仍然存在相关文献遗漏的情况.因此,本文对初步接受的相关文献进行了滚雪球,以找出相关文献的参考文献中那些进行了类似工作的研究,并成功找出两篇相关文献.文献的另一个有效性威胁是灰色文献被纳入研究,由于目前对于微服务组织方面的研究并不广泛和深入,相关内容的白色文献数量十分有限.因此,为了补充数据量以及使分析结果更加全面,本文需要加入灰色文献.同时,在计算机领域,很多研究人员积极地通过博客和问答论坛等形式分享经验和技巧.Zhang 等人[14]在一项软件工程领域使用灰色文献的调查中发现:灰色文献已经在软件工程的研究中表现出了巨大的潜力,正在成为软件工程研究重要的组成部分.在本文的研究过程中,灰色文献确实为我们开发思路和寻找证据等方面提供了很大的帮助.并且,灰色文献也需要经过选择和排除过程、全文阅读和质量评估,与研究相关且高水平的灰色文献才会被接收.因此,灰色文献带来的有效性威胁被尽可能地降低.

5.2 定性分析方法

定性分析方法并不如定量分析方法那么容易被接受,因为分析的过程涉及到作者的主观性(subjectivity),而这会对研究结果的有效性构成威胁.随着越来越多的与研究相关的定性数据需要被综合,人们逐渐认识到定性研究的重要性.它不是定量数据的补充,而是数据综合的一种单独方法.在软件工程中,有大量的定性数据,特别是在本文的研究中,数据大多是定性的,所以使用定性的研究方法是合适且有价值的.同时,为了尽可能地减少作者的主观性差异,每一条三阶解释都进行了评估,合理的解释会被保留,作者意见存在分歧的三阶解释会被拒绝或通过不断修改直至成为合理的解释.

5.3 开发组织适应性评估框架

尚未开始迁移和已经迁移到微服务的公司可以使用评估框架来确定其开发组织对微服务体系结构的适应性,并根据框架中的建议改进其适应性.虽然评估框架在设计时尽可能做到涵盖全面和提出的建议有效,且本文通过问卷和访谈两种形式验证了框架的有效性,但仍然缺乏指导实际的微服务项目中构建团队的案例,我们计划在下一步工作中结合实际项目来验证该框架.

5.4 问卷和访谈

本文使用问卷和访谈这两种调查工具来验证所提出的开发组织适应性评估框架的有效性.为了尽可能减少结果的有效性威胁,本文在计划调查、设计调查、质量评估、执行调查等各个阶段都采取了极其严谨的态度.在基于问卷的设计中,最常见的威胁之一是对问题和回答所采用的措辞不当.为了尽可能减少这种威胁,本文不断地对问卷进行定义和细化,并且经过一位在问卷设计方面有丰富经验的专家的指导进行了多次修改.此外,在问卷的设计中,本文还进行了多次内部研究者和随机挑选的外部参与者的试点测试,以精炼对问题的措辞.同时,本文的问卷参与者和受访的专家均是由本文严格挑选和邀请的国内正在使用微服务架构的知名企业中的人员,以保证数据来源尽可能可靠和有效,而受访者不同程度的经验也确保了本文研究的普遍性.

6 结论和未来的工作

微服务架构以其灵活的可伸缩性、高可维护性和持续交付等优势,吸引了越来越多的关注和研究.但是微服务并非是一颗银弹(silver bullet),不是所有的公司和软件项目都可以使用微服务进行开发.同时,想要成功地迁移向微服务和获取到使用微服务的预期收益,就必须创建一个与之相适应的开发组织.因此,为了帮助使用微服务的公司跟上架构的变化,构建与微服务相适应的开发组织,设计出一套全面且有效的适应性评估框架是必不可少的.而本文所做工作正是为了解决这一需求,同时希望通过本文的工作,吸引更多的微服务研究人员参与到微服务开发组织的研究中.

本文专注于讨论微服务架构给组织方面带来影响的研究,对微服务给开发组织带来的影响进行了一次系统文献综述.我们发现:相关的白色文献(19 篇)数量并不多,特别是在中文检索库中,几乎很难找到符合研究标准的关于组织方面研究的中文文献.为了使系统文献综述的结果更加全面和可靠,我们将高质量的灰色文献(13 篇)纳入了本文的研究.灰色文献已经在软件工程的研究中显示出了成为重要组成部分的潜力[19],及时可靠的灰色文献有助于研究工作并促进了软件工程领域的发展.同时,在本文研究中纳入的白色文献和灰色文献都经过了质量评估,以确保数据的可靠性和结果的有效性.

通过系统文献综述的过程,本文得出了使用微服务架构给组织带来影响的 7 个方面,分别是组织结构、自治团队、技术/工具、组织文化、开发人员、DevOps 和通信,并且明确了这些方面对开发组织产生的影响.而基于系统文献综述的结果,使用元-民族志(步骤 4~步骤 6)对定性数据进行了合成,通过一阶解释和二阶解释形成的论据线,本文得出了 4 条三阶解释.最终,根据系统文献综述和元-民族志合成数据的结果,本文导出了组织对微服务适应性的评估框架并进行了详细的描述.尚未开始微服务架构迁移和已经完成迁移的公司都可以根据这个评估框架对开发组织对于微服务架构的适应性进行评估,并根据框架中给出的建议提高适应性.

同时,基于开发组织适应性评估框架,本文开展了一次问卷调查和一次行业访谈,两者的结果证明了框架的有效性.

最后,本文分析了研究工作中可能出现的有效性威胁,包括相关文献、定性分析方法、提出的框架和问卷访谈这 4 个方面.在未来的工作中,我们将持续关注微服务对开发组织的影响,并且计划结合具体的微服务项目,使用本文所提出的开发组织适应性评估框架来帮助公司构建团队.

References:

[1] Jonas F, Justus B, Stefan W, Alfred Z. Microservices migration in industry: Intentions, strategies, and challenges. In: Foutse K, ed. Proc. of the Int'l Conf. on Software Maintenance. 2019. 481−490. [doi: 10.1109/ICSME.2019.00081]

[2] James L, Martin F. Microservices: A definition of this new architectural term. 2014. http://martinfowler.com/articles/microservices.html

[3] Conway Melvin E. How do committees invent. Datamation Journal, 1968,14(4):28−31.

[4] Sasa B, Vivian N, Andy K. Architecting microservices: Practical opportunities and challenges. Journal of Computer Information Systems, 2018,60(5):428−436. [doi: 10.1080/08874417.2018.1520056]

[5] Stefan H, Rainer W, Georg B. An expert interview study on areas of microservice design. In: Chao KM, ed. Proc. of the Service Oriented Computing and Applications. Germany: Springer-Verlag, 2018. 137−144. [doi: 10.1109/SOCA.2018.00028]

[6] Justus B, Alfred Z. Towards integrating microservices with adaptable enterprise architecture. In: Nurcan S, ed. Proc. of the Enterprise Distributed Object Computing. 2016. 1−6. [doi: 10.1109/EDOCW.2016.7584392]

[7] Di FP, Patricia L, Ivano M. Migrating towards microservice architectures: An industrial survey. In: Proc. of the 2018 IEEE Int'l Conf. on Software Architecture (ICSA). 2018. 29−39. [doi: 10.1109/ICSA.2018.00012]

[8] Zhang H, Li SS, Jia ZJ, Zhong CX, Zhang C. Microservice architecture in reality: An industrial inquiry. In: Proc. of the 2019 IEEE Int'l Conf. on Software Architecture (ICSA). 2019. 51−60. [doi: 10.1109/ICSA.2019.00014]

[9] Halappa S, Williams R. Getting your team ready for microservices. 2018. https://dzone.com/articles/getting-your-team-ready-for-microservices-part-1

[10] Barbara K, Pearl BO, David B, Mark T, John B, Stephen L. Systematic literature reviews in software engineering—A systematic literature review. Information Software Technology, 2009,51(1):7−15. [doi: 10.1016/j.infsof.2008.09.009]

[11] Zhang C, David B. What do we know about the effectiveness of software design patterns? IEEE Trans. on Software Engineering, 2012,38(5):1213−1231. [doi: 10.1109/TSE.2011.79]

[12] Jacopo S, Damian T, Van DHW. The pains and gains of microservices: A systematic grey literature review. Journal of Systems Software, 2018,146:215−232. [doi: 10.1016/j.jss.2018.09.082]

[13] Outi SK, Sarah B, Ita R. Challenges and recommended practices for software architecting in global software development. Information and Software Technology, 2019,106:234−253. [doi: 10.1016/j.infsof.2018.10.008]

[14] Zhang H, Zhou X, Huang X, Huang H, Ali BM. An evidence-based inquiry into the use of grey literature in software engineering. In: Proc. of the ACM/IEEE 42nd Int'l Conf. on Software Engineering. 2020. [doi: 10.1145/3377811.3380336]

[15] Affan Y, Ijlal HM. On the quality of grey literature and its use in information synthesis during systematic literature reviews [MS. Thesis]. School of Computing, Blekinge Institute of Technology, 2012.

[16] Fu CL, Zhang H, Huang X, Zhou X, Li Z. A review of meta-ethnographies in software engineering. In: Proc. of the Evaluation and Assessment on Software Engineering. 2019. 68−77. [doi: 10.1145/3319008.3319015]

[17] Barbara K, Tore D, Magne J. Evidence-based software engineering. In: Proc. of the Int'l Conf. on Software Engineering. 2004. 273−281. [doi: 10.1109/ICSE.2004.1317449]

[18] Claes W, Per R, Martin H, Ohlsson MC, Björn R, Anders W. Experimentation in Software Engineering: An Introduction. 3rd ed., Springer-Verlag, 2012. 50−51. [doi: 10.1007/978-3-642-29044-2_10]

[19] Orit H, Liora N. Teaching and learning qualitative research≈conducting qualitative research. The Qualitative Report, 2014,19(24): 1−29.

[20] Noblit GW, Dwight HR. Meta-ethnography: Synthesizing Qualitative Studies. 1988. 9−14.

[21] Nicky B, Rona C, Catherine P, Jenny D, Myfanwy M, Roisin P. Using meta ethnography to synthesise qualitative research: A worked example. Journal of Health Services Research Policy, 2002,7(4):209−215. [doi: 10.1258/135581902320432732]

[22] Atkins S, Lewin S, Smith HJ, Engel ME. Conducting a meta-ethnography of qualitative literature: Lessons learnt. BMC Medical Research Methodology, 2008,8(1):21−31. [doi: 10.1186/1471-2288-8-21]

[23] Maggie C, Nicola R, Isabelle U, Edward D, Ruth J, Margaret M, Roberts RJ, Louise TR, Andrew B, Nicky B. Improving reporting of meta-ethnography: The eMERGe reporting guidance. BMC Medical Research Methodology, 2019,19(1): 1−13. [doi: 10.1002/rev3.3147]

[24] Henrik GJ. Enabling autonomous teams in large-scale agile through architectural principles. In: Aguiar A, ed. Proc. of the 19th Int'l Conf. on Agile Software Development Companion. New York: Association for Computing Machinery, 2018. 1−4. [doi: 10.1145/3234152.3234183]

[25] Miika K, Niko M, Tommi M. Challenges When Moving from Monolith to Microservice Architecture, Current Trends in Web Engineering. Cham: Springer-Verlag, 2018. 32−47. [doi: 10.1007/978-3-319-74433-9_3]

[26] Fred B. Adding manpower to a late software project makes it later. In: The Mythical Man-Month: Essays on Software Engineering. 1975.

[27] Florian R, Jonas S, Philip W, Sabine S, Albert Z. Microservice architecture and model-driven development: Yet singles, soon married? In: Kuhrmann M, ed. Proc. of the Int'l Conf. on Agile Software Development. New York: Association for Computing Machinery, 2018. 12−23. [doi: 10.1145/3234152.3234193]

[28] Pooyan J, Claus P, Mendonca NC, James L, Stefan T. Microservices: The journey so far and challenges ahead. IEEE Software, 2018, 35(3):24−35. [doi: 10.1109/MS.2018.2141039]

[29] Davide T, Valentina L, Claus P. Microservices, Science and Engineering. Springlink, 2020. [doi:10.1007/978-3-030-31646-4_5]

[30] Chen LP. Microservices: Architecting for continuous delivery and DevOps. In: Proc. of the 2018 IEEE Int'l Conf. on Software Architecture (ICSA). 2018. 39−46. [doi: 10.1109/ICSA.2018.00013]

[31] Sunil J. Organization & cultural impact of microservices architecture. In: Ahram T, ed. Proc. of the Int'l Conf. on Applied Human Factors and Ergonomics. Los Angeles: Springer Int'l Publishing, 2017. 89−95. [doi: 10.1007/978-3-319-60747-4_9]

[32] Pavel M. Microservices organizational practices. 2019. https://dzone.com/articles/microservices-organizational-practices

[33] O'connor RV, Peter E, Paul C. Exploring the impact of situational context: A case study of a software development process for a microservices architecture. In: Raffo D, ed. Proc. of the Int'l Conf. on Software and Systems Process. 2016. 6−10. [doi: 10.1145/2904354.2904368]

[34] Armin B, Abbas H, Pooyan J. Microservices architecture enables DevOps: Migration to a cloud-native architecture. IEEE Software, 2016,33(3):42−52. [doi: 10.1109/MS.2016.64]

[35] Liu BH, Zhang H, Dong LM. DevOps China research. Ruan Jian Xue Bao/Journal of Software, 2019,30(10):3206−3226 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5797.htm [doi: 10.13328/j.cnki.jos.005797]

[36] Mikko K, Pekka A, Pekka K. A case study on the impact of customer communication on defects in agile software development. In: Proc. of the Agile Conf. 2006. 76−88. [doi: 10.1109/AGILE.2006.1]

[37] Saonee S, Suprateek S. Exploring agility in distributed information systems development teams: An interpretive study in an offshoring context. Information Systems Research, 2009,20(3):440−461. [doi: 10.1287/isre.1090.0241]

[38] Richard HJ, Oldham GR. Motivation through the design of work: Test of a theory. Organizational Behavior and Human Performance, 1976,16(2):250−279. [doi: 10.1016/0030-5073(76)90016-7]

[39] Valentina L, Outi SK. On the negative impact of team independence in microservices software development. In: Aguiar A, ed. Proc. of the 19th Int'l Conf. on Agile Software Development Companion. New York: Association for Computing Machinery, 2018. 1–4. [doi: 10.1145/3234152.3234191]

[40] Justus B, Jonas F, Stefan W, Alfred Z. Microservices in industry: Insights into technologies, characteristics, and software quality. In: Proc. of the 2019 IEEE Int'l Conf. on Software Architecture Companion (ICSA-C). 2019. 187−195. [doi: 10.1109/ICSA-C.2019.00041]

[41] Davide T, Valentina L, Claus P. Processes, motivations, and issues for migrating to microservices architectures: An empirical investigation. In: Proc. of the IEEE Int'l Conf. on Cloud Computing Technology and Science. 2017. 22−32. [doi: 10.1109/MCC.2017.4250931]

附中文参考文献:

[35] 刘博涵,张贺,董黎明.DevOps 中国调查研究.软件学报,2019,30(10):3206−3226. http://www.jos.org.cn/1000-9825/5797.htm [doi: 10.13328/j.cnki.jos.005797]

附录 1:白色文献和灰色文献的质量评估表

Table 1-1 Quality assessment of white literature

表 1-1 白色文献质量评估

评估项	取值(Y/P/N)
是否对研究的背景有适当的描述?	–
研究的目的是否有明确的说明?	–
研究设计是否符合研究目的?	–
是否对研究使用的研究方法进行介绍?	–
数据收集过程的结构设计是否合理?	–

Table 1-1 Quality assessment of white literature (Continued)

表 1-1 白色文献质量评估(续)

评估项	取值(Y/P/N)
检索的数据质量是否有保证?	–
数据分析是否足够严格?	–
结论是否准确回答了研究问题?	–
对于研究结果是否有明确的表述?	–
是否对有效性的威胁进行了讨论并提供对这些威胁的处理?	–

Table 1-2 Quality assessment of grey literature

表 1-2 灰色文献质量评估

评估项	取值(Y/P/N)
是否对研究的背景有适当的描述?	–
研究方法是否被说明?	–
文章的逻辑是否清晰以及文章结构是否有条理?	–
文章的结论是否清晰?	–
文章中是否能提取出帮助我们回答研究问题的数据?	–
文章是否包含有关书目足够的信息?(作者、发表日期等)?	–
是否被其他文献引用过?	–
这篇文章在网上的论坛或讨论页面是否有评论?	–
如果在网上讨论,这些评论对原始文章的发现有贡献吗?	–
作者是否属于学术/研究机构?	–

附录 2:被选取文献的编号和基本信息

Table 2-1 Code and basic information of selected literature

表 2-1 被选取文献的编号和基本信息

文献编号	题目	作者	年份
S1	Organization & Cultural Impact of Microservices Architecture	Sunil Joshi	2017
S2	On the Negative Impact of Team Independence in Microservices Software Development	Valentina Lenarduzzi and Outi Sievi-Korte	2018
S3	Enabling Autonomous Teams in Large-scale Agile Through Architectural Principles	Jan Henrik Gundelsby, Knowit, Lakkegata, Oslo and Norway	2018
S4	Do the Microservices Improve the Agility of Software Development Teams?	Branislav Mišić, Milana Novković, Robert Ramač and Vladimir Mandić	2017
S5	Microservices Architecture Enables DevOps: Migration to a Cloud-native Architecture	Armin Balalaie, Abbas Heydarnoori and Pooyan Jamshidi	2016
S6	Architecting Microservices: Practical Opportunities and Challenges	Saša Baškaradaa, Vivian Nguyenb and Andy Koroniosa	2018
S7	Microservices: Architecting for Continuous Delivery and DevOps	Lianping Chen	2018
S8	Challenges When Moving from Monolith to Microservice Architecture	Miika Kalske, Niko M¨akitalo and Tommi Mikkonen	2017
S9	Towards Integrating Microservices with Adaptable Enterprise Architecture	Justus Bogner and Alfred Zimmermann	2016
S10	Challenges and Recommended Practices for Software Architecting in Global Software Development	Outi Sievi-Kortea, Sarah Beecham and Ita Richardson	2019
S11	Migrating towards Microservices: Migration and Architecture Smells	Andrés Carrasco, Brent van Bladel and Serge Demeyer	2018
S12	Drivers and Barriers for Microservice Adoption— A Survey among Professionals in Germany	Holger Knoche and Wilhelm Hasselbring	2019
S13	Microservice : The Journey So Far and Challenges Ahead	Pooyan Jamshidi, Claus Pahl, Nabor C. Mendonca and James Lewis	2018
S14	Microservices in a Small Development Organization— An Industrial Experience Report.	Georg Buchgeher, Mario Winterer, Rainer Weinreich, Johannes Luger, Roland Wingelhofer and Mario Aistleitner	2017
S15	Exploring the Impact of Situational Context— A Case Study of A Software Development Process for a Microservices Architecture	Rory V. O'Connor, Peter Elger and Paul M. Clarke	2016

Table 2-1 Code and basic information of selected literature (Continued)

表 2-1 被选取文献的编号和基本信息(续)

文献编号	题目	作者	年份
S16	Microservice Architecture: Aligning Principles, Practices, and Culture	Irakli Nadareishvili, Ronnie Mitra, Matt McLarty and Mike Amundsen	2016
S17	Microservices Anti-Patterns: A Taxonomy	Davide Taibi, Valentina Lenarduzzi and Claus Pahl	2019
S18	(g) Service Per Team	Chris Richardson	2019
S19	(g) Microservices and Team Organization	Thomas Jardinet	2017
S20	(g) How to Manage Agile Teams with Microservice	Suri Patel	2019
S21	(g) Microservices Organizational Practices	Pavel Micka	2019
S22	(g) Getting Your Team Ready for Microservices	Sowmya Halappa and RJ Williams	2018
S23	(g) Who Runs Microservices DevOps	Andrew Smith	2019
S24	(g) Microservices and Conway's Law	Darrell Burgan	2017
S25	Microservices in Agile Software Development a Workshop Based Study into Issues Advantages and Disadvantages	Davide Taibi, Valentina Lenarduzzi, Claus Pahl and Andrea Janes	2017
S26	Microservice Migration in Industry: Intentions, Strategies and Challenges	Jonas Fritzsch, Justus Bogner, Stefan Wagner and Alfred Zimmermann	2019
S27	(g) Organizational Structure, Microservices and Conway's Law	Serhiy Masyutin	2018
S28	(g) The Size of a Microservice is the Size of the Team that is Building It	Aviran Mordo	2016
S29	(g) How Much Freedom Should Be Given to Microservice Teams?	Kelly Goetsch	2018
S30	(g) 3 Tips for Moving Your Team to a Microservices Architecture	Jake Lumetta	2018
S31	(g) Six Challenges Every Organization Will Face Implementing Microservices	Jonas Fritzsch, Justus Bogner, Stefan Wagner and Alfred Zimmermann	2019
S32	(g) 成功的微服务,需要企业组织架构如何变革?	网易云	2019

附录 3:完成的网格:对组织的影响

Table 3-1 Finished grid: Impact on the organization

表 3-1 完成的网格:对组织的影响

概念	组织结构	自治团队	技术/工具	组织文化	开发人员	DevOps	通信	主要思想/结论(二阶解释)
S1	组织规模、关系	–	–	文化成为一个大话题	–	–	团队内部和团队间的沟通	"探讨有效的微服务文化中的关键因素,包括组织规模、动机、关系以及它们如何影响微服务的发展和目标"
S2	–	无须等待其他团队做出决定	–	–	–	–	减少团队间的通信需求	与单体系统相比,微服务开发需要较少的团队间通信
S3	团队和组织结构	高度的自主权	使用不同的工具和技术	–	不知道这个微服务是做什么	–	–	使用微服务架构可以增加团队自治性
S4	团队结构改革	–	自由决定使用哪种编程语言	–	缺乏开发经验	–	–	与其他架构相比,没有经验的团队成员在开始使用微服务时需要较少的工作
S5	团队结构需要改变	独立的团队	使用不同开发语言	–	团队需要熟悉这些概念的成员	微服务支持DevOps	–	DevOps 和微服务之间的协作将有助于消除开发和运营团队之间的障碍
S6	围绕计划构建和运行筒仓组织	–	异构技术堆栈的可能性	需要组织信任	需要重要的高级技能	DevOps	–	总结了 19 个在微服务中没有考虑到的因素,并表明并不是所有类型的公司都能从微服务中获益
S7	组织结构是重要因素	团队可以自主进行开发、发布和操作	不受控制的方式使用不同的技术堆栈会导致问题	组织文化是重要因素	团队没有足够的领域知识和经验	DevOps	缩短客户到负责的开发人员的沟通路径	在 DevOps 和 CD 环境中,使用微服务可以带来很多优势

Table 3-1 Finished grid: Impact on the organization (Continued 1)

表 3-1 完成的网格:对组织的影响(续 1)

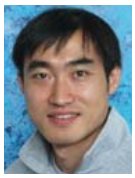
概念	组织结构	自治团队	技术/工具	组织文化	开发人员	DevOps	通信	主要思想/结论 (二阶解释)
S8	要求改变 团队的 组成	团队可以 决定该服务 内部发生什么	更自由的 选择工具	–	拥有相应 技能的人 可能不够多	需要采用 DevOps 文化	微服务减 少了通信 需求	当软件公司足够大时, 微服务是处理复杂性 和规模的好方法
S9	–	–	技术多样性的 激增可能 很快变得 难以控制	DevOps 文化	–	结合 DevOps 元素	–	“致力于将微服务与 可适应的 企业体系结构集成”
S10	架构与 组织结构 之间缺乏 一致性	–	–	–	–	–	通信开销 被最小化	微服务中的组织, 工作方式, 设计实践,模块化 和任务分配
S11	单层团队 异味	–	–	–	没有 经验的 开发人员	–	–	提出了 9 个常见的 微服务陷阱 及其可能的解决方案
S12	–	高度的团队 自主性和 高度的责任感	团队自主 选择工具	文化 需要变革	开发者 缺乏技能	DevOps 的 适应性	–	开发和操作人员的技能 被认为是主要障碍.
S13	–	增加了团队 未能看到 全局的风险	重用和 故障处理 变得困难	这些问题 必须作为 组织文化的 一部分来处理	–	–	–	“微服务架构 还有很长的 路要走,在这个过程中 还存在许多挑战”
S14	–	负责有关 其服务的 所有决策	–	–	开发人员 必须学习 每一种 技术	–	–	公司可以在使用 微服务中,从技术级 获益,但需要在 组织级进行调整
S15	–	–	–	团队文化 对变化的 抵抗力较低	员工需要 非常高的 核心技术 能力	–	–	探索情景上下文 对微服务 开发的影响
S16	–	–	–	文化无形 但重要	–	–	–	“调整原则、实践和 文化的微服务 体系结构”
S17	–	–	–	–	没有 经验的 开发人员	–	–	“根据从业者在开发 基于微服务的 系统时所经历的不良 实践确定了一组 (20 个)微服务反模式”
S18	–	自治和 松散耦合	–	–	–	–		一个团队应该只拥有 一个服务,这足以确保 团队自治和松耦合
S19	组织需要 与架构 匹配	–	–	开发人员 对变革的 抵制	–	–	–	组织结构需要改变, 文化需要得到尊重 但也必须进行变革
S20	组织团队	–	–	–	–	–		“如何使用 微服务管理团队”
S21	团队数量 增加	–	不再有义务 使用 相同的技术	组织 文化 评估	–	DevOps	改进 沟通 渠道	“了解部署微服务的 主要优势,以及采用 基于微服务的产品 所必需的公司文化”
S22	–	帮助实现 自治	–	–	–	–	缺乏 沟通	介绍使用单体架构开发 应用程序相关问题以及 如何适应微服务架构
S23	首先建立 和组织团队	-	–	开发文化 发生改变	–	投资 DevOps	-	在微服务架构上使用 DevOps 的优势, 加速了开发和部署等

Table 3-1 Finished grid: Impact on the organization (Continued 2)

表 3-1 完成的网格:对组织的影响(续 2)

概念	组织结构	自治团队	技术/工具	组织文化	开发人员	DevOps	通信	主要思想/结论(二阶解释)
S24	组织如何演变以适应	–	–	–	–	–	–	给出了使用微服务架构组织需要如何进行演变的建议
S25	–	独立部署	–	–	需要高技能的开发人员	–	寻找不同的沟通策略	“采用微服务仍然是复杂的,从单一系统迁移成功取决于几个因素”
S26	–	自治团队间的协作	–	文化更加开放和不受约束	–	DevOps 实践加强了团队精神	–	“对迁移到微服务的背景下的意图、策略和挑战进行了定性研究”
S27	架构影响结构	–	–	–	–	–	–	conway 定律与组织结构的关系
S28	微服务的大小是构建它的团队的大小	团队解耦	任何技术堆栈的选用	–	–	DevOps	–	微服务不能解决所有问题,同时它与 DevOps 密切相关
S29	–	所有权和责任下放到每个团队	更多的自由去选择技术	–	–	–		公司需要在给团队充分自治权与控制之间取得正确的平衡
S30	组织结构影响文化	–	灵活的技术选择	管理团队文化	给成员成长机会	–	–	团队迁移到微服务的技巧:成功的团队文化,给每个团队成员一个成长的机会和合理的技术选择
S31	组织结构反映架构	-	–	内在缺乏信任	–	–	–	提出了使用微服务时组织会遇到的 6 个问题
S32	合理调整组织架构	–	–	–	–	DevOps 化	–	“组织需要走向去中心化、小团队化、全能化和 DevOps 化”

崔海涛(1997－),男,硕士生,CCF 学生会员,主要研究领域为微服务架构,经验软件工程.



章程(1982－),男,博士,副教授,CCF 专业会员,主要研究领域为微服务,软件体系结构,经验软件工程.

丁翔(1998－),男,硕士生,CCF 学生会员,主要研究领域为微服务架构,经验软件工程.

曹伶俐(1995－),女,硕士生,CCF 学生会员,主要研究领域为微服务架构,经验软件工程.

杨耘(1963－),男,博士,教授,博士生导师,主要研究领域为云计算,边缘计算,服务计算,软件技术,工作流.