**Software Cybernetics**

1. K. Y. Cai, T. Y. Chen, T. H. Tse. Towards Research on Software Cybernetics, 2002.

调查研究软件控制论这一新兴领域的潜力。

1. K. Y. Cai, J. W. Cangussu, R. A. Decarlo. An overview of software cybernetics, 2003.

软件控制论的主题和思想及综述。

介绍目前存在的一些研究课题：软件过程中的反馈机制、双模性和可控性（Bisimulation and Controllability）、适应性软件、软件综合、软件测试过程控制、适应测试。

潜在研究课题：适应性软件复兴和软件性能控制、软件容错的控制理论方法、控制系统的逻辑基础、控制系统的共享（community）复杂性。

1. J. W. Cangussu, R.A. DeCarlo, A.P. Mathur. Using sensitivity analysis to validate a state variable model of the software test process, 2003.

在所有被研究的案例中，状态变量模型(Model S)的表现都是基于经验的用于验证模型的观察结果。提出了使用敏感性分析来验证软件测试过程的状态变量模型，结果还表明，当模型表现不符合预期时，敏感性分析可以对模型进行结构改进。

1. K. Y. Cai, X.Y. Wang. Towards a control-theoretical approach to software fault tolerance, 2004.

现有的软件容错方案是基于冗余和多样性的思想。有效但缺乏严格的理论基础。增加了软件的复杂性，导致很高的开发成本，给实时并发软件系统带来了挑战，在这些系统中，时序需求可能非常严格，并发进程中的错误可能会相互传播。

本文将软件容错问题视为一个鲁棒监控（robust supervisory control，RSC）问题，提出了RSC方法。

1. P Lin, A MacArthur, J Leaney. Defining autonomic computing: a software engineering perspective, 2005.

自主计算是开发大规模分布式系统的一种很有前途的新方法。本文提出对自主计算系统的普遍接受的定义。

1. Scott D. Miller Raymond A. DeCarlo Aditya P. Mathur A control-theoretic approach to the managementof the software system test phase, 2006.

为软件系统测试阶段(STP)的模型编写一个定量的、适应性的过程控制技术。该技术将参数校正和模型预测控制相结合，克服了建模错误、参数估计错误和对可用于生产力改进的资源的限制所带来的问题

1. Requirement process establishment and improvement from the viewpoint of cybernetics, 2006.

用控制理论定义了一个需求工程（RE）过程控制系统，它的动态和稳态性能，以及系统设计、分析和改进的步骤。它强调了与度量元素相关的过程活动的需求，包括反馈补偿和组织支持。

1. F Belli, K. Y. Cai, R. Decarlo. Introduction to the special section on software cybernetics, 2006.

综述性文章介绍软件控制论的一些成果（以上）。

1. P. Wang, K.Y. Cai. Representing extended finite state machines for SDL by a novel control model of discrete event systems, 2006.

本文讨论了SDL的EFSM，并将其转化为一种新的离散事件系统控制模型。

1. S. S. Yau, D Huang, L Zhu, K. Y. Cai. A software cybernetics approach to deploying and scheduling, 2007.

许多分布式系统正在采用基于服务的系统(SBS)。SBS中的应用程序通常可以看作是遵循特定工作流的各种计算服务的组合。这些工作流通常需要满足各种时间和资源约束。本文提出了一种利用时间和资源约束来部署和调度SBS中工作流的软件控制论方法。在我们的方法中，开发了一种基于逻辑的技术，用于建模和解决SBS中工作流的时间和资源约束，以生成工作流的初始资源分配、调度和部署计划。应用软件控制论的原理和概念来指导软件控制器的综合，以监控和适应系统的行为

1. M Huebscher, J McCann. A survey of autonomic computing - degrees, models, and applications, 2008.

自主计算将许多计算领域结合在一起，目的是创建自我管理的计算系统。以一个已建立的参考模型的组成部分为例，我们讨论了为该领域做出重大贡献的工作，讨论组成自治系统的大型系统，这些系统说明了其体系结构的层次结构性质。

1. C Liu, C Jiang, H Hu, K. Y. Cai, D Huang, S. S. Yau. A control-based approach to balance services performance and security for adaptive service-based systems (ASBS), 2009.

本文提出了一种基于控制的方法来平衡适应性基于服务的系统(ASBS)的安全性和性能之间的平衡。对提供不同安全级别的加密算法的加密/解密延迟和内容大小之间的关系进行了测量和建模。

1. J Chen, Q Zhang, S. D. Bruda. Cybernetics in software system verification, 2009.

本文将控制理论中的负反馈概念应用于软件系统的验证。为了实现软件系统的验证，我们研究了软件测试、模型检验以及它们与反馈的结合。讨论了控制论与软件工程相互作用的未来工作。

1. O.S. Donaires. Programming in the complex: Cybernetic insights into software process and architecture, 2010.

本文提出了一个系统控制论过程模型，它是由斯塔福比尔可行系统模型(VSM)和贝瑞伯姆螺旋模型组成的。VSM提供了对软件体系结构的适应性和对软件过程的自组织能力。

1. T Gao, T Li, Z Xie, J Xu. A process model of software evolution requirement based on feedback, 2011.

为提高软件进化的效率和质量，本文对软件演化需求过程进行了建模。通过对软件演化、软件需求和反馈机制的研究，提出了软件演化需求过程中的两种不同类型的反馈，完善了软件演化需求的变更控制过程。

1. B Gaudin, A Bagnato. Software maintenance through supervisory control, 2011.

本文提出了一种基于控制理论的方法，允许自动生成维护补丁。这种方法禁用有缺陷或脆弱的系统功能，并要求在部署之前对系统进行测试，以便以后可以对系统进行监视，并在运行时与管理程序进行交互。

1. D Ionescu,. Autonomic computing: the path towards controlling cloud computing services 2011.

本文讨论了云服务的各种设计原则和自主计算算法。重点将放在自主计算的实时架构(参考虚拟化和云上计算)、每个云上服务中的本地自主计算（self-organizing algorithms）单元的自组织算法和全球云上服务的适应性策略等问题上

1. Q Yang, J Lü, J Xing, X. Tao, H Hu., Y Zou. Fuzzy control-based software self-adaptation: a case study in mission critical systems, 2011.

任务关键型软件(MCS)尤其需要自适应能力。本文提出了一种基于fuzzy控制的方法，为实现软件自适应提供了一种系统、工程、直观的方法。这种方法使用fuzzy逻辑来处理软件中的不确定性，使用自然语言风格来描述自适应逻辑，并使控制在软件中可见。这大大缩短了控制工程与软件工程之间的知识和语义差距，降低了软件自适应的实现难度。

1. L Wang, Y Gao, C Cao, L Wang. Towards a general supporting framework for self-adaptive software systems, 2012.

面对内部和环境的动态变化，各种软件系统越来越需要自适应能力。上面的远景需要自适应方法来处理这些挑战，本文提出了一个通用的支持框架，用于软件系统自适应运行环境的动态变化，同时满足各种用户需求。该框架涉及三个关键问题:1)整体控制架构，采用双闭环方式，分别包括自适应回路和自学习回路；2)通用描述语言，是表示目标系统自适应知识的与应用无关的统一语言；3)前向推理、规划、反馈强化学习三种实现机制，由上述描述语言支持，运行时在不同模块中执行。

1. V. E. S. Souza. A requirements-based approach for the design of adaptive systems, 2012.

系统由人类管理可能会变得过于复杂，因此，将必须自我管理、自我配置自己的运作、自我保护、自我改正错误、和自我调整。论文提出了面向目标和目标软件密集的社会技术系统（goal-oriented and targets softwareintensive socio-technical systems），试图将控制回路方法与分散代理启发的方法结合起来。

1. R Buyya, R. N. Calheiros, X Li. Autonomic cloud computing: open challenges and architectural elements, 2012.

由于云是复杂的、大规模的、异构的分布式系统，对其资源的管理是一项具有挑战性的任务。因此它们需要自动化和集成的智能策略来提供资源，从而提供安全、可靠和经济有效的服务。因此，对服务的有效管理成为构成云计算结构的软件平台的基础。在这个方向上，本文确定了自主资源供应中的开放问题，并提出了支持云上SaaS应用程序的创新管理技术。我们提出了一个概念性的架构，早期的结果证明了云的自主管理的好处。

1. M Harmana. The role of artificial intelligence in software engineering, 2012.

本篇论文提出了基于搜索的软件工程的最新进展，以及在软件工程的概率推理和机器学习方面的长期工作。本文探讨了这些紧密相关的工作之间关系，认为它们有很多共同点，并为软件工程在人工智能领域提出了一些未来的挑战。

1. M Harman., S. A. Mansouri, Y. Zhang, 2012. Search-based software engineering: trends, techniques, and applications.

在过去的五年中，基于搜索的软件工程(Search-Based Software Engineering, SBSE)的研究工作有了显著的增长，本文对有关SBSE的文献进行了综述和分类。

1. D. B. Abeywickrama, N. Hoch, F. Zambonelli. SimSOTA: engineering and sim- ulating feedback loops for self-adaptive systems, 2013.

实现一个分散式系统自主服务组件有多个,它们之间的相互作用的反馈循环是非常具有挑战性的。虽然有几项工作将反馈循环表示为一级实体，但是很少注意提供实际的工具支持。本文提出了一种基于反馈环的自适应系统设计方法。我们还展示了SimSOTA的第一个实现，这是一个正在开发的Eclipse插件，用于支持基于我们的基于反馈环的方法的自适应系统的建模、模拟和验证。

1. K. Ravindran, M Rabby. Software cybernetics to infuse adaptation intelligence in networked systems, 2013.

本文采用软件控制论方法，智能物理系统模块(IPW)体现了核心的适应功能，以响应不断变化的环境条件和用户输入。IPW在系统的有限操作区域内表现出智能行为。

1. P Mayer. The autonomic cloud: a vision of voluntary, peer-2-peer cloud computing, 2013.

在本文中，我们探讨了以平台即服务的计算基础设施形式的自主云的思想（an autonomic cloud in the form of a platform-as-a-service computing infrastructure），与通常的实践不同，这种基础设施不是由一组维护良好的高性能计算机组成，而是由一组自愿提供的异构节点松散集合组成，这些节点以点对点的方式连接。

1. [H Yang](http://xueshu.baidu.com/s?wd=author%3A%28Hongji%20Yang%29%20&tn=SE_baiduxueshulib_9r82kicg&ie=utf-8&sc_as_para=sc_lib%3Austb&sc_from=ustb&sc_f_para=sc_hilight%3Dperson), [Z Lu](http://xueshu.baidu.com/s?wd=author%3A%28Lu%20Zhang%29%20&tn=SE_baiduxueshulib_9r82kicg&ie=utf-8&sc_as_para=sc_lib%3Austb&sc_from=ustb&sc_f_para=sc_hilight%3Dperson). Controlling and Being Creative: Software Cybernetics and Creative Computing, 2014.

软件控制论强调控制软件，而创造性计算强调在整个软件生命周期内的创造性,控制和创造性几乎是软件开发和进化中最重要的两个方面。本文通过提出控制软件行为和软件开发过程的模型，讨论了如何将这两个方面最好地结合起来进行软件改进。

1. X Zhao, J Xue, C Hu, R Ma, S Zhang. Research on software behavior modeling based on extended finite state automata,2014.

软件行为模型可以用于检测异常行为,验证协议,生成测试用例,捕捉异常事件序列和检查程序的兼容性特性。本文提出了一种基于扩展有限状态自动机的软件行为建模方法，它基于现有的软件行为建模方法，重点关注数据值的约束和软件组件之间的交互轨迹。结合了Daikon和ESC/JAVA工具，得到了扩展有限状态自动机边的约束，从而提高了模型的精度。利用该方法实现了一个软件行为建模系统。实验结果表明，该模型能够获取越来越多的准确信息，为软件分析、验证和测试提供了良好的保证。

1. L Zhang, B.B. Yin, J. Lv, K.Y. Cai, S.S. Yau, J. Yu. A history-based dynamic random software testing, 2014.

DRT中有一些缺陷，例如仅仅根据测试用例是否揭示缺陷来更新测试剖面并不严格。为了克服原DRT的不足，在DRT的基础上提出了基于历史的动态随机测试，记作DRT-h。在DRT-h策略中，利用历史测试信息实时计算软件测试过程中各子域故障检测效率的估计值。In DRT-h strategy, the estimate of the defect detection rate of each sub domain is calculated in real-time during the software testing by using history testing information.

1. K. P. Kumar, N. S. Naik. Self-Healing model for software application, 2014.

本文提出了一种自修复机制，用于监测、诊断和修复应用程序中损坏的文件，使其恢复到原来的状态。应用程序的分析部分将通过在运行时维护相应文件的哈希值并从原始应用程序中恢复损坏的文件来完成。

1. K Ahuja, H Dangey. Autonomic Computing: An emerging perspective and issues, 2014.

概述自主计算的介绍，并讨论一些共同构成自主计算重大挑战的问题和挑战：下一代应用程序、环境和信息基础设施正迅速变得脆弱、难以管理和不安全。这使得越来越复杂的系统必须由高技能的IT专业人员来管理。但是，由于计算环境的复杂性和异构性，人为错误可能会日益妨碍整个环境。因此，引入了一种新的计算范式来处理动态、复杂性和不确定性，这就是所谓的自主计算。

1. Y. Abuseta, K. Swesi. Towards a framework for testing and simulating self- adaptive systems, 2015.

自适应系统被认为是一种很有前途的解决方案，用于管理当今软件密集型系统的复杂性和不确定性。然而，他们的设计还没有系统地进行。特别是，在开发一个框架的问题上几乎没有做什么工作，该框架包含必要的可重用流程元素和系统组件，这些元素和组件用自适应功能来增强开发中的系统，并使系统开发人员能够在部署之前模拟和测试这些功能。本文试图解决这个问题，并提出了一个自适应系统的测试框架。这里只介绍框架的高级需求和概念架构。自适应能力是围绕IBM blueprint提出的反馈控制回路设计的。

1. J Baeten, J Markovski. The role of supervisory controller synthesis in automatic control software development, 2015.

本文讨论了监督控制器合成在自动化软件代码生成中的作用，实现了一个基于模型的系统和软件工程框架。

1. F Alvares, E Rutten, L Seinturier. Behavioural model-based control for autonomic software components, 2015.

自主管理器(Autonomic Managers, AMS)主要用于软件组件内部的自主控制重构。这种管理是基于过去的监视事件、配置以及定义适应逻辑和不变属性的行为程序来执行的。这里的挑战是提供关于在配置空间中导航的保证，这需要做出涉及到对系统可能的未来进行预测的决策。本文提出了基于逻辑离散控制方法的AMs设计。并定义了一种特定于域的语言（Domain Specific Language），名为Ctrl-F，它提供高级构造来描述软件组件上下文中的行为程序。Ctrl-F的正式定义是通过转换为有限状态自动机给出的，该自动机允许通过验证或离散控制器合成自动探索行为程序。

1. J Liu, J Zhou, R Buyya. Software Rejuvenation based fault tolerance scheme for cloud applications, 2015.

本文提出了一种面向云应用的全面软件更新容错方案，该方案包含三个不可或缺的部分:适应性故障检测、老化程度评估和基于跟踪重放的构件更新检查点。通过初步的定性评估，我们的新容错方案对云应用的可用性带来了很好的改善。

1. V. Chang. A Cybernetics social cloud, 2015.

本文提出了一个社交云，给出了系统的设计、开发和分析。这项技术基于BOINC开源软件，所有涉及的步骤都严格遵循了控制论的系统设计、实现、实验和验证，以确保在任何时候都有高质量的输出和服务。这为控制论应对大数据研究的挑战提供了独特的贡献。

1. K. M. Sim. Agent-based interactions and economic encounters in an intelligent InterCloud, 2015.

InterCloud是一个相互连接的全球“云的云”，它使每个云都可以利用其他云的资源。为了促进云间联盟的形成，本文设计了一种新的四阶段云间交互协议和一组云间代理的新策略。

1. E .F. Coutinho, D. G. Gomes. An autonomic computing-based architecture for cloud computing elasticity, 2015.

弹性是云计算的一个重要特性，可以理解为计算云如何通过配置和删除资源来适应其工作负载的变化。自主计算为弹性云计算解决方案的构建带来了许多非常有用的概念，比如控制循环和基于阈值的规则。本文提出了一种基于自主计算概念的云计算弹性体系结构。

1. Z Ding, Z Wei, H Chen. A software cybernetics approach to self-tuning performance of on-line transaction processing systems, 2016.

在线事务处理(OLTP)系统的自调优是一项具有挑战性和耗时的任务，因为在数据库管理系统(dbms)中需要自动配置多个性能参数。在本文中，我们提出了一种软件控制论方法来自调整dbms的性能。设计一个基于fuzzy逻辑的适应性控制DBMS，使它有能力控制对象、性能参数、并更新控制器本身。

1. H Liu, Y Liu, L Liu. The Verification of Program Relationships in the Context of Software Cybernetics, 2016.

.本文将软件控制论的思想应用于验证过程中，建立了一个嵌套控制系统。该方法在用户需求不断变化的动态环境中验证功能需求，以程序为被控对象，以软件行为模型(SBM)确定的验证策略为控制器。

1. L Liu, Q Zhou, J Liu. Requirements cybernetics: Elicitation based on user behavioral data , 2016.

当今基于web的应用程序收集各种服务数据，这是产品设计者更好地了解用户需求和行为的理想信息源。本文首先讨论了目前收集到的数据类型，然后将这种数据驱动的需求捕获过程表示为反馈控制系统，其中经典的需求捕获思想转化为对用户行为模型的持续优化。

1. C Perera, A. V. Vasilakos. A knowledge-based resource discovery for Internet of Things, 2016.

本文提出了一种知识驱动的方法，称为上下文感知传感器配置模型(CASCOM)，以简化配置物联网中间件平台的过程，使数据消费者，特别是非技术人员，能够方便地检索到他们需要的数据。