《企业级 AIOps 实施建议》白皮书

发起单位：高效运维社区

AIOps 标准工作组

指导单位：数据中心联盟

云计算开源产业联盟

发布时间：2018 年 4 月 13 日

发布版本：V0.6

目录

背景介绍

AIOps 即智能运维，其目标是，基于已有的运维数据（日志、监控信息、应用信息等），通过机器学习的方式来进一步解决自动化运维所未能解决的问题，提高系统的预判能力、稳定性、降低 IT 成本，并提高企业的产品竞争力。

Gartner 在 2016 年时便提出了 AIOps 的概念，并预测到 2020 年，AIOps 的采用率将会达到 50%。AIOps 目前在国内外领先的互联网企业开始被逐渐应用，也是近年来国内外被普遍看好的新技术。

为了让国内众多互联网中小企业、特别是传统企业可以共享、复用国内外顶尖互联网的 AIOps 技术和能力，并能够更快捷的进行 AIOps 相关产品选型，因此开展国内外第一个 AIOps 白皮书及相关标准制定工作。

AIOps 标准将分成两大类，分别适用于企业内部的 AIOps 能力建设与评估、及企业购置相关 AIOps 产品的认证评估，使得 AI 真正落地应用于运维，造福于企业。

此白皮书由高效运维社区牵头，为 AIOps 标准工作组成员及所属企业的相关经验汇总。

组织单位

AIOps 白皮书及标准由云计算开源产业联盟（英文译名： Open Source Cloud Alliance for industry ,缩写为：OSCAR）下设 AIOps 标准工作组、数据中心联盟（DCA）下设组织 IT 运维委员会（即开放运维联盟）及高效运维社区联合发起制定，对外以 AIOps 标准工作组开展工作，由萧田国担任组长。

OSCAR 联盟和 DCA 均为中国信息通信研究院牵头、国内相关企事业单位共同发起、在中国通信标准化协会（CCSA）的指导下成立的第三方非盈利组织，负责相关联盟标准及行业标准的制定和推广。

中国信息通信研究院是工业和信息化部直属单位，国家高端专业智库、产业创新发展平台，支撑相关行业发展的重大战略、规划、政策、标准和测试认证等。

中国通信标准化协会（英文译名为：China Communications Standards Association，缩写为：CCSA）于 2002 年 12 月 18 日在北京正式成立。该协会是国内企、事业单位自愿联合组织起来，经业务主管部门批准，国家社团登记管理机关登记，开展通信技术领域标准化活动的非营利性法人社会团体。协会的主要任务是为了更好地开展通信标准研究工作，把通信运营企业、

制造企业、研究单位、大学等关心标准的企事业单位组织起来，把具有我国自主知识产权的标准推向世界，支撑我国的通信产业，为世界通信作出贡献。

IT 运维委员会（开放运维联盟）是中国第一个也是唯一的运维行业协会。

AIOps 标准工作组、IT 运维委员会和高效运维社区的负责人均为萧田国。

编写成员

AIOps 标准工作组当前成员包括来自 BAT、360、京东、华为、中国银行、平安科技、宜信等企业及 AIOps 解决方案提供方的 AIOps 领域专家。

发起人及发起单位

萧田国	高效运维社区（创始人），AIOps 标准工作组（组长）

顾问

裴丹	清华大学（AIOps 实验室 负责人）

编审成员[1]

王哲	360（AIOps 负责人）
许斯亮	360
毛茂德	阿里巴巴
王肇刚	阿里巴巴
刘大鹏	必示科技（联合创始人）
曲显平	百度（AIOps 负责人）
哈晶晶	百度
萧田国	高效运维社区（创始人）
周荣	华为（消费者 BG AIOps 负责人）
陶仕敏	华为
孙培	华为
王超	京东金融
孙熠青	宜信（高级副总裁）
张真	宜信（AIOps 负责人）

[1]说明：按公司首字母序

朱品燕　　灵犀（创始人）
陈亚殊　　平安科技（AIOps 负责人）
刘洋　　平安科技
裴丹　　清华大学 副教授 青年千人
屈中泠　　擎创科技
饶琛琳　　日志易
郑华贵　　数智慧（创始人）
刘栖铜　　腾讯 IEG（AIOps 负责人）
党受辉　　腾讯 IEG
涂彦　　腾讯 IEG
胡飞雄　　腾讯 IEG
岳磅　　腾讯 IEG
赵建春　　腾讯 SNG（AIOps 负责人）
张戎　　腾讯 SNG
刘扬清　　中国银行

本版本核心编写成员

赵建春　　腾讯 SNG（AIOps 负责人）
张戎　　腾讯 SNG
周荣　　华为（消费者 BG AIOps 负责人）
孙培　　华为
刘栖铜　　腾讯 IEG（AIOps 负责人）
胡飞雄　　腾讯 IEG
曲显平　　百度（AIOps 负责人）
郑华贵　　数智慧（创始人）　//实施及关键技术模块
饶琛琳　　日志易　　//实施及关键技术模块
屈中泠　　擎创科技　　//实施及关键技术模块
陶仕敏　　华为 2012 实验室（资深 AIOps 专家）

1、整体介绍

AIOps，即 Artificial Intelligence for IT Operations，智能运维，将人工智能应用于运维领域，基于已有的运维数据（日志、监控信息、应用信息等），通过机器学习的方式来进一步解决自动化运维没办法解决的问题。

早期的运维工作大部分是由运维人员手工完成的，这被称为手工运维或人肉运维。这种落后的生产方式，在互联网业务快速扩张、人力成本高企的时代，难以维系。

自动化运维因此应运而生。其基于用可被自动触发的、预定义规则的脚本，来执行常见的、重复性的运维工作，从而减少人力成本，提高运维效率。总的来说，自动化运维可以认为是一种基于行业领域知识和运维场景领域知识的专家系统。

随着整个互联网业务急剧膨胀，以及服务类型的复杂多样，“基于人为指定规则”的专家系统逐渐变得力不从心。自动化运维的不足，日益凸显。

DevOps 的出现，部分解决了上述问题。其强调从价值交付的全局视角，端到端打通软件生命周期，建立基于微服务的单件流式的流水线。但 DevOps 更强调横向融合及打通，较低阶段的 DevOps 无力改变“基于认为指定规则”的既定事实。

AIOps 是 DevOps 在运维（技术运营）侧的高阶实现，两者并不冲突。此部分可具体参考《研发运营一体化能力成熟度模型》。

AIOps 不依赖于人为指定规则，主张由机器学习算法自动地从海量运维数据（包括事件本身以及运维人员的人工处理日志）中不断地学习，不断地提炼并总结规则。

AIOps 在自动化运维的基础上，增加了一个基于机器学习的大脑，指挥监测系统采集大脑决策所需的数据，做出分析、决策，并指挥自动化脚本去执行大脑的决策，从而达到运维系统的整体目标。

AIOps 基于自动化运维，将 AI 和运维很好的结合起来，其需要三方面的知识：

1) 行业领域知识：应用的行业，如互联网、金融、电信、物流、能源电力、工业制造和智慧城市等，并熟悉生产实践中的难题；
2) 运维场景领域知识：如指标监控、异常检测、故障发现、故障止损、成本优化、容量规划和性能优化等；
3) 机器学习：把实际问题转化为算法问题，常用算法包括如聚类、决策树、卷积神经网络等。

AIOps 和 DevOps 两者并不冲突，企业级 DevOps 涵括包括运维在内的整个软件生命周期，AIOps 是 企业级 DevOps 在运维（技术运营）侧的高阶实现。

AIOps 是运维的发展必然，是自动化运维的下一个发展阶段。Gartner 相关报告预测 AIOps 的全球部署率将从 2017 年的 10%增加到 2020 年的 50%。其应用行业，除了互联网以外，还包括高性能计算、电信、金融、电力网络、物联网、 医疗网络和设备、航空航天、军用设备及网络等领域。

本白皮书综合国内领先的互联网公司、金融企业及 AIOps 解决方案提供方的相关经验，给出了一种企业级 AIOps 的 AIOps 理论方法和生产实践，希望能帮助贵司快速、成功实施 AIOps。

本白皮书聚焦 AI 应用到 Ops 领域，不涉及自动化运维相关内容。

2、AIOps 目标

AIOps，通俗的讲，是对规则的 AI 化，即将人工总结运维规则的过程变为自动学习的过程。具体而言，是对我们平时运维工作中长时间积累形成的自动化运维和监控等能力，将其规则配置部分，进行自学习的“去规则化”改造，最终达到终极目标：“有 AI 调度中枢管理的，质量、成本、效率三者兼顾的无人值守运维，力争所运营系统的综合收益最大化”。

AIOps 的目标是，利用大数据、机器学习和其他分析技术，通过预防预测、个性化和动态分析，直接和间接增强 IT 业务的相关技术能力，实现所维护产品或服务的更高质量、合理成本及高效支撑。

3、AIOps 能力框架

AIOps 的建设可以先由无到局部单点探索、再到单点能力完善，形成解决某个局部问题的运维 AI“学件”，再由多个具有 AI 能力的单运维能力点组合成一个智能运维流程。

AIOps 能力框架基于如下 AIOps 能力分级。AIOps 能力分级可具体可描述为 5 级（图-2）：

1) 开始尝试应用 AI 能力，还无较成熟单点应用
2) 具备单场景的 AI 运维能力，可以初步形成供内部使用的学件
3) 有由多个单场景 AI 运维模块串联起来的流程化 AI 运维能力，可以对外提供可靠的运维 AI 学件
4) 主要运维场景均已实现流程化免干预AI 运维能力，可以对外提供可靠的AIOps 服务。
5) 有核心中枢 AI，可以在成本、质量、效率间从容调整，达到业务不同生命周期对三个方面不同的指标要求，可实现多目标下的最优或按需最优。

图 3-1 AIOps 能力分级

学件，亦称 AI 运维组件，类似程序中的 API 或公共库，但 API 及公共库不含具体业务数据，只是某种算法，而 AI 运维组件（或称学件），则是在类似 API 的基础上，兼具对某个运维场景智能化解决的“记忆”能力，将处理这个场景的智能规则保存在了这个组件中。

这个智能规则是在一定量的数据下学习而来的，且具有“可重用”，“可演进”，“可了解”的特性，既可共享由专家利用数据训练的算法，又可保护数据和隐私。

“学件”（Learnware）一词由南京大学周志华老师原创，学件（Learnware）= 模型（model）+规约（specification），具有可重用、可演进、可了解的特性。

很多人可能在自己的应用中已经建立了类似的模型，他们也很愿意找到一个地方把这些模型分享出去。这样一来，一个新用户想要应用，也许不用自己去建立一个，而是先到“学件”市场上找一找有没有合适的，拿来直接或修改后使用。学件基于专家基础上建立，所以比较容易得到专家级的结果，又因为共享出来的是模型，所以避免了数据泄露和隐私泄露的问题。

基于上述 AIOps 能力分级，对应的 AIOps 能力框架如下。

图 3-2 AIOps 能力框架

相关关键运维场景的 AIOps 演进如下。

能力等级	能力描述	Operational Tasks（运维任务）	部署变更场景	故障处理场景	容量管理场景	服务咨询场景	Execution（命令执行）	Perception（需求、服务状态理解）	Planning（规划）	Proactive Learning（主动学习）
1	简单智能化	部分动作取代人	基于模版的部署动作	基于规则的检测、分析、止损动作	基于规则的调度、扩缩容动作	基于规则匹配的应答动作	人+系统	人	人	人
2	单场景的智能化	部分场景取代人	部署、生效、检查等场景智能化	异常检测、根因分析、止损等场景智能化	数据分析、流量调度、扩缩容等场景智能化	自然语言理解、查询、应答等场景智能化	系统	人+系统	人	人
3	多场景协同智能化	复杂场景取代人	智能触发、部署、生效、检查等可协同	智能异常检测、故障诊断、止损等可协同	智能采集、分析、调度、扩缩容等可协同	自然语言理解、知识库、应答等可协同	系统	系统	人+系统	人
			关键点：决策规划由运维系统做出，而不是人 人负责：制定优化目标（比如，可用性、效率、成本等） 运维系统负责：根据其对待处理的需求、待解决的问题的理解，以及对运维对象的认知（经验），自主做出解决方案（规划）并在控制执行过程中根据目标和运维对象的状态反馈来适时调整执行规划							
4	高度智能化	大部分场景无人值守	可自主规划上线方案的智能无人值守变更	可自主规划止损方案的智能故障ONCALL	可自主规划成本和容量方案的智能调度&伸缩	基于意图理解和多轮对话的智能客服	系统	系统	系统	人为主系统辅助
5	完全智能化	完全无人值守	支撑服务完整生命周期的全部基础运维工作，由智能运维系统接管，并不借助人来应对服务和环境的变迁，自主做到可用性、成本、效率的最优化				系统	系统	系统	系统

图 3-3 关键运维场景的 AIOps 演讲

[2]

- “可重用”的特性使得能够获取大量不同的样本；
- “可演进”的特性使得可以适应环境的变化；
- “可了解”的特性使得能有效地了解模型的能力。

4、AIOps 平台能力体系

AIOps 工作平台能力体系主要功能是为 AIOps 的实际场景建设落地而提供功能的工具或者产品平台，其主要目的是降低 AIOps 的开发人员成本，提升开发效率，规范工作交付质量。AIOps 平台功能与一般的机器学习（或者数据挖掘）平台极为类似，此类产品国外的比如 Google 的 AutoML（https://cloud.google.com/automl/）。

图 4-1 AIOps 平台功能模块

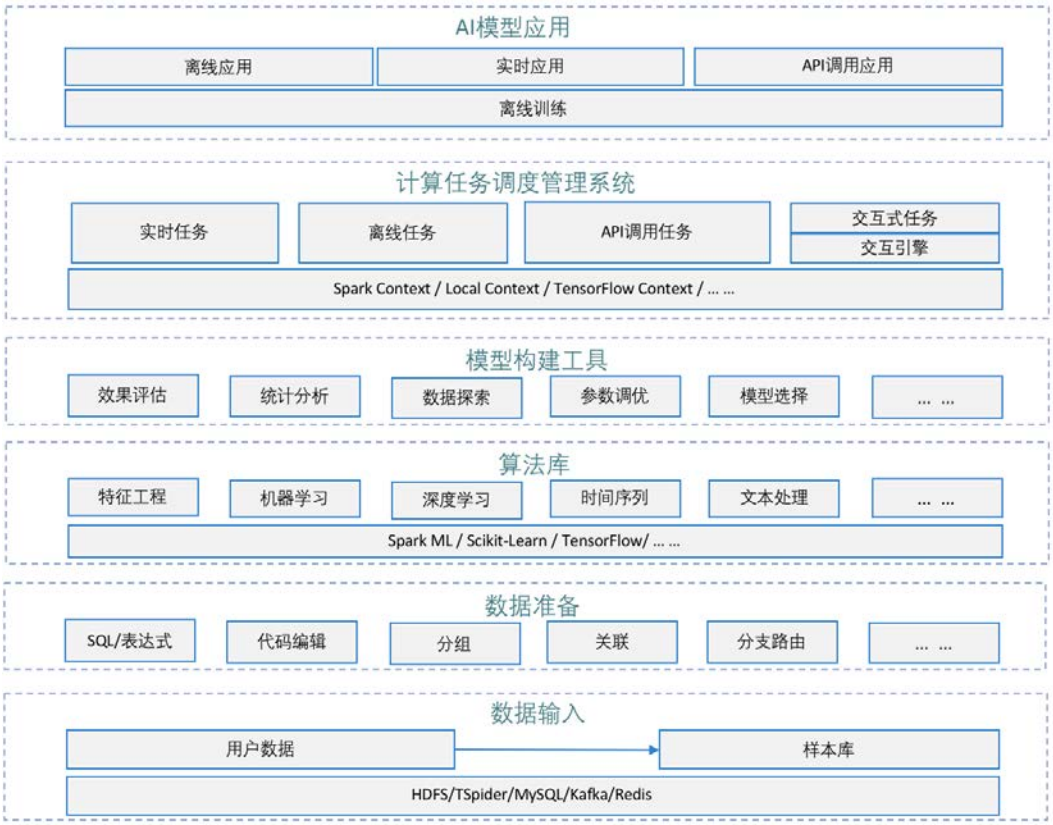

图 4-2 AI 建模服务能力

如上图 4-1、图 4-2，具体的工具或者产品应具备以下功能或模块：

1) 交互式建模功能：该功能支持用户在平台上交互式的进行模型的开发调试，通过简单的方法配置完成模型的构建。
2) 算法库：用户可以在算法库中找到常见常用的算法直接使用，算法按照用途分类，以供用户方便的使用。
3) 样本库：样本库用于管理用户的样本数据，供用户建模时使用，支持样本的增删改查等基本操作。
4) 数据准备：该功能支持用户对数据进行相关的预处理操作，包括关联、合并、分支路由、过滤等。
5) 灵活的计算逻辑表达：在基本常用的节点功能之外，用户还需要自由的表达一些计算逻辑，该需求主要是通过让用户写代码或表达式来支持。
6) 可扩展的底层框架支持：平台本身要能够灵活的支持和兼容多种算法框架引擎，如 Spark、TensorFlow 等，以满足不同的场景以及用户的需求。
7) 数据分析探索：该功能是让用户能够方便快捷地了解认识自己的数据，用户只有基于对数据充分的认识与理解，才能很好的完成模型的构建。

8） 模型评估：对模型的效果进行评估的功能，用户需要依据评估的结论对模型进行调整。

9） 参数以及算法搜索：该功能能够自动快速的帮助用户搜索算法的参数，对比不同的算法，帮助用户选择合适的算法以及参数，辅助用户建模。

10）场景模型：平台针对特定场景沉淀的解决方案，这些场景都是通用常见的，用户可以借鉴参考相关的解决方案以快速的解决实际问题

11）实验报告：模型除了部署运行，相关挖掘出来的结论也要能够形成报告，以供用户导出或动态发布使用。

12）模型的版本管理：模型可能有对个不同的版本，线上运行的模型实例可能分属各个不同的版本，版本管理支持模型不同版本构建发布以及模型实例版本切换升级等。

13）模型部署应用：模型构建完成后需要发布应用，模型部署应用功能支持模型的实例化，以及相关计算任务的运行调度管理。

14）数据质量保障：全链路的数据监控，能够完整的掌控数据的整个生命周期，具备对丢失的数据执行回传补录的能力，保障数据的可用性。

5、 AIOps 团队角色

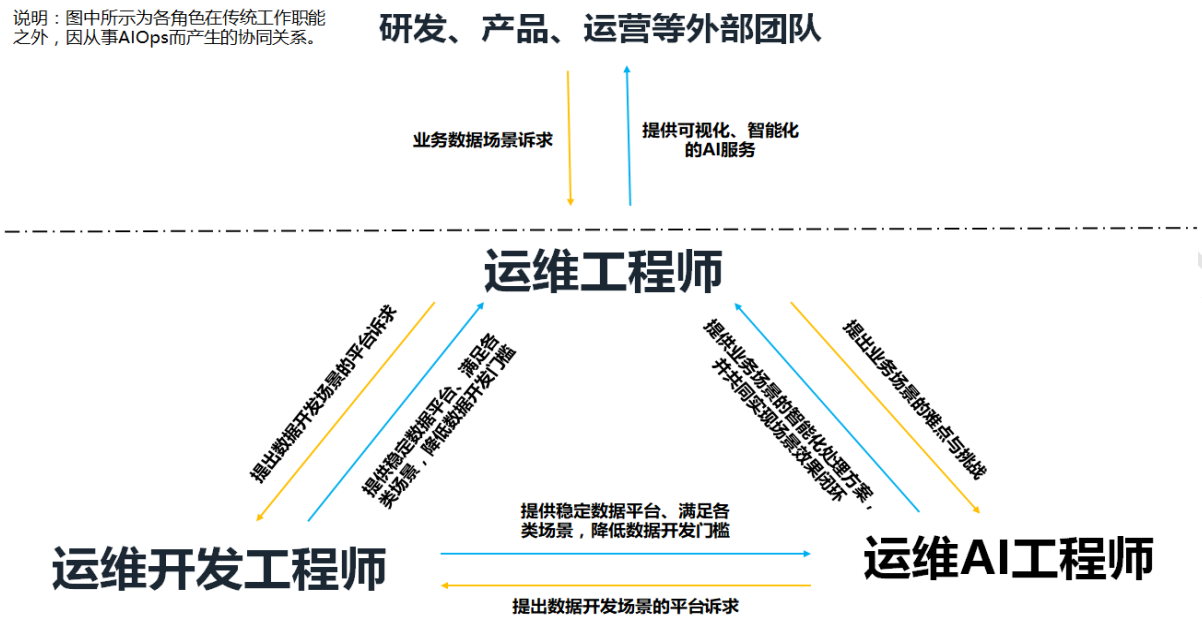

图 5-1 AIOps 团队角色及和外部的协同关系

AIOps 团队内部人员根据职能可分为三类团队，分别为运维工程师团队、运维开发工程师团队和运维 AI 工程师团队，他们在 AIOps 相关工作中分别扮演不同的角色，三者缺一不可。

5.1 运维工程师

能从业务的技术运营中，提炼出智能化的需求点。在开发实施前能够考虑好需求方案，规范数据格式。前期可以通过仿真手法探索和验证方案可行性，起草合适的算法方案。

5.2 运维开发工程师

负责进行平台相关功能和模块的开发，以降低用户使用门槛，提升用户使用效率，并且将运维数据工程师交付的数据通过友好的方式展现给用户。

根据企业 AIOps 程度和能力的不同，运维开发工程师中的运维自动化平台开发和运维数据平台开发的权重不同。

5.3 运维 AI 工程师

针对来自于运维工程师和算法方案进行理解和梳理，完成最终落地方案的输出工作；在工程落地上能够考虑好健壮性、鲁棒性、敏捷性等，合理拆分任务，保障成果落地，以提升最终业务运营质量。

6、AIOps 常见应用场景

AIOps 围绕质量保障、成本管理和效率提升的基本运维场景，逐步构建智能化运维场景。在质量保障方面，保障现网稳定运行细分为异常检测、故障诊断、故障预测、故障自愈等基本场景；在成本管理方面，细分为指标监控，异常检测，资源优化，容量规划，性能优化等基本场景；在效率方面，分为智能预测，智能变更、智能问答，智能决策等基本场景（注：三者之间不是完全独立的，是相互影响的，场景的划分侧重于主影响维度）。

无论是效率提升，质量监控，还是成本优化，都离不开最基础的数据采集，它是整个 AIOp 的基石。 AIOps 提高运维生产力的一种方式就是把质量处理流程中的人力部分尽可能的都替换成机器来做。在机器的分析过程中，系统运行过程中的每一个部件都需要数据支持。无论是海量数据采集、还是数据提取方面都离不开大数据技术。

从数据采集的层面来看，运维数据的采集往往是实时的，数据采集端需要具备一定分析能力，综合考虑用户流量、隐私，服务器压力等多个因素，尽可能的降低无效数据的采集，增加有价值信息的上报。

从数据提取的层面来看，运维的数据是多样化的，历史数据，流数据，日志数据、网络数据、算法数据、文本和 NLP 文档数据，以及 APP 数据、浏览器数据、业务系统运营指标数据等，从这些海量的数据中提取出正真有价值的指标化数据并可视化是进一步分析决策的前提条件。

而成本优化和效率的提升同样离不开数据的支撑。例如，开始实施成本优化的 AIOPS 前，需要尽可能多的收集目前的服务器，网络设备，应用服务，数据库等的性能信息，应用日志信息，tracing 信息，以便对成本优化的效果进行评估。例如，在搭建智能客服机器人的时候，就需要提供充足的问题库和相应的答案才能够建立好一个较优的模型。

图 6-1 AIOps 常见应用场景枚举

以下为各个方向应用场景的能力描述。

	效率提升方向	质量保障方向	成本管理方向
第一阶段（尝试应用）	在这个阶段，尝试在变更，问答，决策，预测领域使用人工智能的能力，但是并没有形成有效的单点应用，这个阶段可以聚焦于数据采集和可视化	在这个阶段，没有成熟的单点应用，主要是手动运维、自动化运维和智能运维的尝试阶段，这个阶段可以聚焦于数据采集和可视化	在这个阶段，运维的成本管理方向还在尝试引入人工智能，但是并没有成熟的单点应用，这个阶段可以聚焦于数据采集和可视化
第二阶段（单点应用）	在这个阶段，在一些小的场景下，人工智能已经可以逐步发挥自己的能力，包括智能变更，智能问答，智能决策，智能预测	在这个阶段，在一些单点应用的场景下，人工智能已经开始逐步发挥自己的能力，包括指标监控，磁盘，网络异常检测等	在这个阶段，在一些小的场景下，人工智能已经开始逐步发挥自己的能力，包括成本报表方向，资源优化，容量规划，性能优化等方向
第三阶段（串联应用）	在这个阶段，人工智能已经将单点应用中的一些模块串联起来，可以结合多个情况进行下一步的分析和操作	在这个阶段，人工智能已经将第二阶段（单点应用）中的一些模块串联在一起，可以综合多个情况进行下一步的分析和操作，包括多维下钻分析寻找故障根因等方向	在这个阶段，人工智能已经将单点应用中的一些模块串联在一起，可以根据成本、资源、容量、性能的实际状况进行下一步的分析和操作
第四阶段（能力完备）	在这个阶段，人工智能能力完备，已经可以基于实际场景实现性能优化，然后进行预测，变更，问答，决策等操作	在这个阶段，人工智能已经基于故障的实际场景实现故障定位，然后进行故障自愈等操作。比如根据版本质量分析	在这个阶段，人工智能的能力已经完备，能够实现基于成本和资源的实际场景实现成本的自主优化，然后进行智能改进的操作

		推断是否需要版本回退，CDN 自动调度等	
第五阶段（终极 AIOps）	在这个阶段，人工参与的成分已经很少，性能优化等整个流程由智能大脑统一控制，并由自动化和智能化自主实施	在这个阶段，人工参与的部分已经很少，从故障发现到诊断到自愈整个流程由智能大脑统一控制，并由自动化和智能化自主实施	在这个阶段，人工参与的成分已经很少，从成本报表方向，资源优化，容量规划，性能优化性等整个流程由智能大脑统一控制，由自动化自主实施

表 6-1 常见应用场景的分类分级能力概述

6.1 效率提升方向

运维效率的提升是运维系统的主要目标之一，自动化运维带来的核心价值之一就是效率提升，而 AIOps 会推动运维效率提升到一个新的高度。其本质的原因是自动化运维依然是人+自动化工具的模式，人工决策与实施依然是主要驱动力，但人会受到自身生理极限以及认知局限的限制，无法持续地面向大规模、高复杂性的系统提供高质量的运维效率。而 AIOps 系统通过深度洞察能力为运维提供持续的，高质量的效率运转。

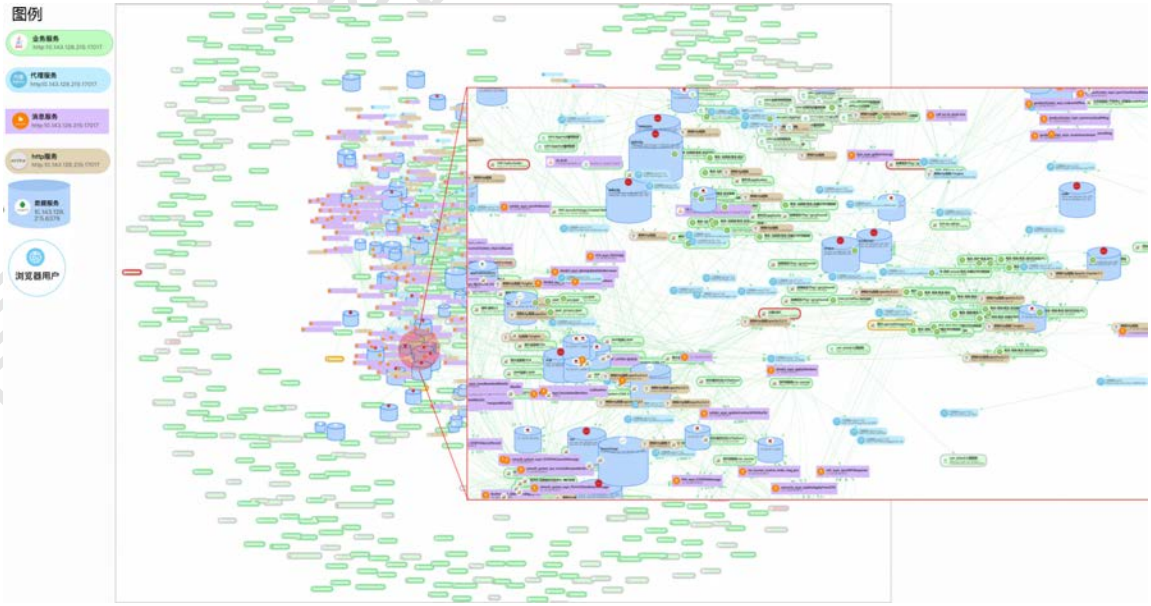

图 6-2 举例（大规模、高复杂性的系统运维，超越人+工具模式的承载力）

图 6-3 效率提升方向的常见应用场景

质量保障是运维的基本场景之一，随着业务的发展，运维系统也在不断的演进，其规模复杂度、变更频率非常大，技术更新也非常的快，与此同时，软件的规模、调用关系、变更频率也在逐渐增大。在这样背景下，需要 AIOps 提供精准的业务质量感知、支撑用户体验优化、全面提升质量保障效率。

6.1.1 智能变更

变更是运维中的一种常见场景，DevOps 通过串联变更的各个环节形成流水线提升了效率，而 AIOps 不仅为变更流水线的各个环节引入了“系统决策”，也能更加持续地，精确地提供高效的变更质量管理。智能变更的系统决策来源于运维人员的运维经验，这些经验通过机器学习，知识图谱等手段转化成系统可学习和实施的数据模型。

AIOps 的智能变更可以应对以下场景：

1) 频繁变更，高速发布的场景：运维人员会由于生理极限以及认知的局限难以应付这样的场景。例如，每天从 1 到 10 次变更时，运维人员通过自动化运维系统尚可应对，如果由 10 次升级到 100 次，甚至更多，就难以高效的，准确的应对了。AIOps 可以根据每次变更的目标，状态，上下文在变更过程中及时做出系统决策，帮助加速变更过程以及规避变更可能带来的问题。

2) 大规模并行变更：随着微服务架构的普及，实际上服务节点会成倍增长，原有几个或几十个节点，可能变成几千甚至上万的规模。人工驱动工具的模式不但受制于人的精力而被迫“串行化”，也制约了变更过程的监察以及变更结果验证的准确性。AIOps 则可以并行驱动更大规模的变更过程，而且变更监察以及结果验证都会被更准确的完成。

6.1.2 智能问答

运维的目标是为了支持稳定，可靠的业务运行，而业务与业务之间既可能有相似性，又可能有差异性。但由于知识背景和对业务的认知差异，往往出现以下情况：

1) 不同的业务人员或开发人员往往会询问运维人员一些相似的问题，运维人员的答案也是非常类似的，但人力被重复消耗。
2) 面对同一个问题，运维人员的回答可能会出现差异（例如表达方式，措辞等），缺乏标准化，可能造成误解。

AIOps 智能问答系统通过机器学习，自然语言处理等技术来学习运维人员的回复文本，构建标准问答知识库，从而在遇到类似问题的时候给出标准的，统一的回复。这样，不仅可以有效地节省运维人员的人力成本，还能够使得提问得到更加及时的回复。

6.1.3 智能决策

许多运维管理工作都需要各种各样的决策，包括扩容，缩容，制定权重，调度，重启等内容。那么可能面临如下问题：

1) 运维人员可以根据自己的业务经验制定相应的决策。但是，不同的业务有着各自的特点，不同的运维人员也有着自己的业务经验。如何将运维人员的这些经验有效地传承是个问题。
2) 人的认知局限性，运维场景的复杂性可能导致最有经验的运维人员遗漏掉某些“不起眼”的“重要细节”，显然，准确的决策还依赖足够充足的细节。

AIOps 智能决策一方面可以将运维人员的决策过程数据化，构建决策支持知识库，从而实现经验积累；另一方面，由于系统掌握了从全局到细节的数据，再结合决策支持知识库，可以为更加准确的决策提供最有力的支撑。

6.1.4 容量预测

运维工作不仅仅包含对当下的决策和处理，往往还需要根据业务的诉求对未来做出合理的规划，包括扩容的预测，缩容的预测等。由于对未来的规划时常存在不确定性，那么规划过程往往需要大量的数据来支持，还需要大量的推演来确定。而人工预测的方式，一方面需要投入大量人力，另一方面运维人员的能力可能存在差异，使得推演的结果品质不尽一致。

AIOps 智能预测借助大数据和机器学习能力，结合运维人员的有效评估经验，甚至业务发展模式以及政策等，对目标场景实现高效的推演过程，最终使预测结果趋近合理范围。这样一

来，不但是人力得以节省，关键在于由于预测效率的提升，使得过去难以重复，耗时耗力的人工预测过程，变得可以应需而变，不断修正预测结果，最终使业务诉求获得最佳预测收益。

6.2 质量保障方向

质量保障是运维的基本场景之一，随着业务的发展，运维系统也在不断的演进，其规模复杂度、变更频率非常大，技术更新也非常的快，与此同时，软件的规模、调用关系、变更频率也在逐渐增大。在这样背景下，需要 AIOps 提供精准的业务质量感知、支撑用户体验优化、全面提升质量保障效率。

图 6-4 质量保障方向常见应用场景

6.2.1 异常检测

运维系统中常见的两大类监控数据源是：指标和文本。前者通常是时序数据，即包含指标采集时间和对应指标的值；后者通常是半结构化文本格式，如程序日志、Tracing 等。随着系统规模的变大、复杂度的提高、监控覆盖的完善，监控数据量越来越大，运维人员无法从海量监控数据中发现质量问题。智能化的异常检测就是要通过 AI 算法，自动、实时、准确地从监控数据中发现异常，为后续的诊断、自愈提供基础。异常检测的常见任务包括对数据源的异常检测，保证数据质量，以及对指标和文本的异常检测。

数据源异常检测：数据源会因为一些不可避免的原因存在一些异常数据，这些异常数据占比虽然很低，但是往往会引起整个指标统计值的波动，使得统计结果偏离真实的用户体验。AIOps 需要自动、实时的动态设置阈值，去除数据源中的异常数据干扰，并能够区分系统真正发生异常时候的故障数据和数据源本身的异常数据，这种判断依赖于一些外部信息。

指标异常检测：包括单指标异常检测及多指标异常检测。其中，单指标异常检测：时间序列指标的异常检测是发现问题的核心环节，传统的静态阈值检测为主的方式，阈值太高，漏告警多，质量隐患难以发现，阈值太低，告警太多引发告警风暴，干扰业务运维人员的判断。AIOps 通过机器学习算法结合人工标注结果，实现自动学习阈值、自动调参，提高告警的精度和

召回率，大幅度降低人工配置成本。其中，多指标异常检测：运维过程中有些指标孤立来看可能并没有异常，但是综合多个指标来看，可能就是异常的。有些单指标表现是异常的，但是综合多个指标来看可能又是正常的。AIOps 需要能够综合多个指标综合评判系统指标异常，提高告警的准确性。

文本异常检测：文本日志常是在特点条件下触发生成的，并遵循一定的模板，即半结构化文本。传统的日志检测有两种方式：1、根据日志级别（如 Info、Warning、Critical）进行报警，但由于其设定不准确，或不满足实际需要，导致准确性差；2、通过设置规则，匹配日志中特定字符串进行报警，但该方法依赖依赖人工经验，且只能检测已知和确定模式的异常。AIOps 需要通过自然语言处理、聚类、频繁模式挖掘等手段，自动识别日志出现的反常模式；结合人工反馈和标注，不断进行优化、完善。

6.2.2 故障诊断

异常检测实现了运维人员对数据的感知，有了数据之后，智能分析可以进一步解放运维人力，提高运维效率，故障诊断是智能分析的核心部分，主要包括基于人工故障库的故障诊断和基于数据挖掘的故障诊断。

基于人工故障库的故障诊断：日常运维过程中，运维人员积累了大量的人工经验，运维过程中的大部分故障都是重复的、人工能够识别的异常。重复问题的定位浪费了大量的人力，而且人工确认过程往往是比较滞后的。AIOps 把人工专家经验固化下来，对常见问题实现分钟级内自动诊断，运维人员收到的告警信息中，就需要包括故障定位的结果信息。

基于数据挖掘的故障诊断：人工经验可能存在偏差，人工认为的原因可能并不是问题的根因，当有些故障首次发生没有人工经验可以借鉴的时候，故障根因也难以定位。尤其随着微服务的发展，业务的组网变得更加复杂，模块多带来的消息路由多、依赖多，问题的定界定位分析更为困难，人工故障决策效率挑战巨大。 对于已知故障，AIOps 能够综合故障数据和人工经验自动提取故障特征，生成故障特征库，自动匹配，自动定位故障；对于未知故障，AIOps 需要根据故障特征推演出可能的故障原因，并在人工确认后加入的故障特征库。

6.2.3 故障预测

故障的出现一般不是突然的，就比如网络故障来说，往往从丢包开始到网络不可用是有一个演变的过程，依据海恩法则：每一起严重事故的背后，必然有 29 次轻微事故和 300 起未遂先兆以及 1000 起事故隐患，开展主动健康度检查，针对重要特性数据进行预测算法学习，提前

诊断故障，避免服务受损；常见场景：磁盘故障预测、网络故障预测（根据交换机日志的交换机故障预测），内存泄露预测等等。

6.2.4 故障自愈

智能分析实现了故障的诊断和预测，智能执行根据智能分析的结果实现故障自愈。传统的故障自愈的决策主要靠人的经验，人的经验能够覆盖的故障范围是有限的，而且人工无法保证 7*24 随时可以立即决策与处理。AIOps 能够提供完善的自动化平台，在故障智能分析之后，自动决策，实现自愈，常见场景：版本升级回退，DNS 自动切换，CDN 智能调度，智能流量调度等。

故障自愈是根据故障诊断的结果的输出（问题定位和根因分析），进而进行影响评估，决定“解决故障”或“恢复系统”的过程。影响评估是对故障之后所产生的影响范围（系统应用层面，业务执行层面，成本损失层面等等）输出评估结果，并根据这个评估来决定要采用什么解决手段，甚至生成解决手段的执行计划。

6.3 成本管理方向

每个公司的经营都离不开成本管理，成本管理包括成本核算，成本分析，成本决策，成本控制。本文不对财务上的成本管理做过多的阐述，主要从 AIOps 方向上在成本分析和决策中能发挥的作用来举例说明。AIOps 通过智能化的资源优化，容量管理，性能优化实现 IT 成本的态势感知、支撑成本规划与优化、提升成本管理效率。 •

图 6-5 成本管理方向的常见应用场景

6.3.1 成本优化

在成本优化方向，需要采取高可用的设计，提供更加合理的服务，包括接入层，业务层，存储层等。在接入层需要提供合理的健康检查机制，更加智能的负载均衡算法，限定流量等工作。在业务层不仅需要去除 DB 的强依赖，使用合理的降级，还要进行合理的压测，监控以

及动态的负载均衡。在存储层需要做的事情是容灾等关键工作。这样的话，可以使得内部数据的质量得到大量提升，外部数据的优先接入和动态选择。

对于设备采集的周期控制这个问题来说，过晚的设备采购可能会影响到业务的正常上线或扩展，而过早的采购也可能造成成本的浪费。于是，AIOps 需要建立合理的模型并建立更好的规划，并据此计算更准确的设备采购计划，也能对成本进行更好的控制。

6.3.2 资源优化

公司的运营成本优化项目一直是公司成本预算的关键一步。优化问题包括设备的优化，带宽，码率的优化等等。只有进行了合理的资源优化，才能够使得公司的成本得到有效的控制。不同的服务的资源消耗类型是不一样的，包括计算密集型，包括存储密集型等等，而对于同一个服务在不同的时间点资源消耗也是不一样的。

对于一个企业来说，识别不同服务的资源消耗类型，识别每个服务的资源瓶颈，实现不同服务间的资源复用是降低成本的重要环节。根据资源应用的性能指标，可以大致分类成以下类别：

1) 计算密集型：CPU 使用率较高，常见于需要大量计算资源的搜索，推荐，数学计算等场景中；
2) 内存密集型：占用的内存使用率较高，如缓存服务；
3) IO 密集型：网络 IO 繁忙或者磁盘 IO 操作繁忙，常见于爬虫，消息管道，分布式存储等服务中。

大型互联网公司里动辄上千上万的应用数，很容易有应用因为业务变化已经访问量不断缩减甚至已经下线，但是线上还占用着大量的资源，通过对应用的性能指标分析，筛选出各项性能指标都很低的应用，就可以识别出这些“被遗忘” 的应用，就可以跟业务负责人进行核对进行缩容或者下线。

目前大部分公司都已经使用了虚拟化或者 docker 技术，同一个物理机上的不同虚拟机或容器已经进行了很好的细粒度资源分配和隔离，所以对于同一台物理机可以进行混合部署不同类型的应用，如计算密集型应用，存储密集型应用，IO 密集型应用混部在同一台物理机上，以提高更大的资源利用率，甚至一定量的“超卖”（通过共享部分资源，实现分配的总的资源数超过物理机的资源数）。对于一些灵活的计算任务，如 Spark，Storm 等计算类任务，还可以使用按时分配的策略，如白天运行在部分服务器上，而且夜间需要运行大批量计算的报表等任务时，利用业务应用夜间资源使用率低的特点，把部分任务分配到业务应用所在的服务器上运行，充分利用这些业务应用的服务器的计算资源，提高整体利用率。

AIOps 通过密度管理、特性管理、碎片管理、木桶管理等方法，优化企业不同服务器的配比，发现并优化资源中的短板，提供不同服务的混合部署建议，最终实现智能化降成本方案分析服务。

6.3.3 容量规划

对于一个企业来说，容量的需求和业务的发展紧密相关。为了保障产品的正常运营，就需要对容量进行合理的预估。如果容量预留过多，则会造成资源浪费；反之，如果容量预留过少，则容易引发现网故障。而传统的基于业务运维人员人工经验容量预测手段不是十分有效，甚至大多数是“拍脑袋”的结果。不准确的容量预估也使得运维缩容和扩容显得被动。

通常来说，大型的互联网公司都会有规模庞大的服务器集群，业务规模增加，新业务上线，过保机器替换都会导致有大量新采购的机器需要上线并扩容到集群中，对于一些特殊场景，如电商网站的大促活动，社交类网站的热点新闻事件等，容量规划更是一件必不可少的考验。活动之后资源往往又需要进行回收缩容操作，以节省运行的成本。

以往的容量规划往往是靠人工经验来操作，现今 AIOps 将根据业务目标的需求，结合服务数据，整合运维人员的业务经验，建立精准容量规划模型，从而精确预测各个业务的容量，让其使用率达到最优。

6.3.4 性能优化

性能的调优一直是运维的重要一环。如果性能优化得当，则会减少实际的运算量，减少内存方面的滥用，提升服务器的性能。运维人员在其中并不能保证及时发现所有潜在的性能问题，很多时候也不知道什么的系统配置才是最优的系统配置，什么时候的权重配比才能够达到最佳的效果。AIOps 能够根据现网的实际情况，进行智能地调整配置，智能发现性能优化策略，提供智能化的优化服务。

7、AIOps 实施及关键技术

为了实现成本管理、效率提升、质量保障的场景，根据 Gartner 的定义，AIOps 产品或平台应包含下图所示的要素：

1) 数据源：大量并且种类繁多的 IT 基础设施
2) 大数据平台：用于处理历史和实时的数据
3) 计算与分析：通过已有的 IT 数据产生新的数据，例如数据清洗、去除噪声等
4) 算法：用于计算和分析，以产生 IT 运维场景所需要的结果
5) 机器学习：这里一般指无监督学习，可根据基于算法的分析结果来产生新的算法

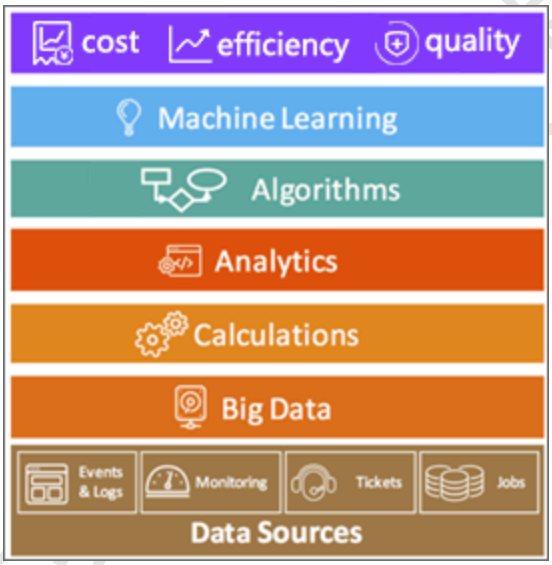

图 7-1 AIOps 产品或平台要素图 说明[3]

7.1 数据采集

数据采集负责将智能运维所需要的数据接入至 AIOps 平台，所接入的运维数据类型一般包括(但不限于)日志数据，性能指标数据，网络抓包数据，用户行为数据，告警数据，配置管理数据，运维流程类数据等。

数据采集方式可分为无代理采集以及有代理采集两种。其中无代理采集为服务端采集，支持 SNMP，数据库 JDBC，TCP/UDP 监听，SYSLOG，Web Service，消息队列采集等主流采集方式。有代理采集则用于本地文件或目录采集，容器编排环境采集，以及脚本采集等。

说明[3] 本图来源 https://www.bmc.com/blogs/what-is-aiops/ 并增加了最上行

7.2 数据处理

针对采集数据进行入库前的预处理，数据从非结构化到结构化的解析，数据清洗，格式转换，以及数据（如性能指标）的聚合计算，处理工作主要体现在几个方面：

1) 数据字段提取：通过正则解析，KV 解析，分隔符解析等解析方式提取字段
2) 规范化数据格式：对字段值类型重定义和格式转换
3) 数据字段内容替换：基于业务规则替换数据字段内容，比如必要的数据脱敏过程，同时可实现无效数据、缺失数据的替换处理
4) 时间规范化：对各类运维数据中的时间字段进行格式统一转换
5) 预聚合计算：对数值型字段或指标类数据基于滑动时间窗口进行聚合统计计算，如取 1 分钟 CPU 平均值

7.3 数据存储

数据存储是 AIOps 平台的数据落地的地方，可以根据不同的数据类型以及数据的消费和使用场景，可选择不同的数据存储方式。数据主要可分为如下几类：

1) 数据需要进行实时全文检索，分词搜索。可选用主流的 ElasticSearch 引擎
2) 时间序列数据（性能指标），主要以时间维度进行查询分析的数据， 可选用主流的 rrdtool、graphite、influxdb 等时序数据库
3) 关系类数据，以及会聚集在基于关系进行递归查询的数据可选择图数据库
4) 数据的长期存储和离线挖掘以及数据仓库构建，可选用主流的 Hadoop、Spark 等大数据平台

7.4 离线和在线计算

离线计算：针对存储的历史数据进行挖掘和批量计算的分析场景，用于大数据量的离线模型训练和计算，如挖掘告警关联关系，趋势预测/容量预测模型计算，错误词频分析等场景。

在线计算：对流处理中的实时数据进行在线计算，包括但不限于数据的查询、预处理和统计分析，数据的实时异常检测，以及部分支持实时更新模型的机器学习算法运用等。主流的流处理框架包括：Spark Streaming, Kafka Streaming, Flink, Storm 等。

7.5 面向 AIOps 的算法技术

运维场景通常无法直接基于通用的机器学习算法以黑盒的方式解决，因此需要一些面向 AIOps 的算法技术，作为解决具体运维场景的基础。有时一个算法技术还可用于支撑另外一个算法技术。 常见的面向 AIOps 的算法技术包括：

1) 指标趋势预测：通过分析指标历史数据，判断未来一段时间指标趋势及预测值，常见有 Holt-Winters、时序数据分解、ARIMA 等算法。该算法技术可用于异常检测、容量预测、容量规划等场景。
2) 指标聚类: 根据曲线的相似度把多个 KPI 聚成多个类别。该算法技术可以应用于大规模的指标异常检测：在同一指标类别里采用同样的异常检测算法及参数，大幅降低训练和检测开销。常见的算法有 DBSCAN, K-medoids, CLARANS 等，应用的挑战是数据量大，曲线模式复杂。
3) 多指标联动关联挖掘: 多指标联动分析判断多个指标是否经常一起波动或增长。该算法技术可用于构建故障传播关系，从而应用于故障诊断。常见的算法有 Pearson correlation, Spearman correlation, Kendall correlation 等，应用的挑战为 KPI 种类繁多，关联关系复杂。
4) 指标与事件关联挖掘: 自动挖掘文本数据中的事件与指标之间的关联关系（ 比如在程序 A 每次启动的时候 CPU 利用率就上一个台阶）。该算法技术可用于构建故障传播关系，从而应用于故障诊断。常见的算法有 Pearson correlation, J-measure, Two-sample test 等，应用的挑战为事件和 KPI 种类繁多，KPI 测量时间粒度过粗会导致判断相关、先后、单调关系困难。
5) 事件与事件关联挖掘: 分析异常事件之间的关联关系，把历史上经常一起发生的事件关联在一起。该算法技术可用于构建故障传播关系，从而应用于故障诊断。常见的算法有 FP-Growth, Apriori，随机森林等，但前提是异常检测需要准确可靠。
6) 故障传播关系挖掘：融合文本数据与指标数据，基于上述多指标联动关联挖掘、指标与事件关联挖掘、事件与事件关联挖掘等技术、由 tracing 推导出的模块调用关系图、辅以服务器与网络拓扑，构建组件之间的故障传播关系。该算法技术可以应用于故障诊断，其有效性主要取决于其基于的其它技术。

说明：

本文档为第一次发布，更多内容如 AIOps 实践路径建议、AIOps 效果度量等内容，因时间关系未能呈现，将在后续版本中予以更新。

如您对本白皮书有任何意见或建议，或者您也想为白皮书贡献一份力量，请联系@东辉，邮箱：yangdonghui@greatops.net，微信：Bohebohe2017，或直接联系高效运维社区其他同事。

本白皮书的线上讨论区为：http://www.gaowei.vip/m/sns-69278531，相关二维码如下。

附录：案例

案例 1：海量时间序列异常检测的技术方案

作者：赵建春，张戎 @腾讯 SNG

1、案例陈述

在很多企业内部，工程师都会收集指标类的监控数据，也就是根据时间来采集相应的指标值。例如某款 APP 的在线用户数，某个场景下的成功数和失败数。随着时间的迁移，整个系统会越来越复杂，监控的数据量会变得越来越大，就会形成海量的时间序列。在这种情况下，运维人员很难通过人工巡查的方式来查看所有的时间序列是否出现了异常，运维人员也无法通过配置规则的方式来解决海量时间序列异常检测的问题。而且，在公司的人力成本有限的情况下，通过人工巡检的方式也无法及时和有效地发现时间序列的异常。

为了解决这类问题，我们针对腾讯 SNG 内外部的场景建设了基于机器学习的时间序列异常检测方案。结合织云 Monitor 监控的具体场景，我们构建了全方位的时间序列异常检测方案。同时，基于腾讯 SNG 丰富的数据集，已经实现了百万条时间序列用少量时间序列检测模型就可以实现异常检测的能力。

2、海量时间序列异常检测的常见问题与解决方案

【常见问题】

在海量时间序列的异常检测中，通过人工巡检的方式明显不足以及时发现时间序列的异常告警。在海量时间序列的异常检测中，通过人工配置规则的方式，针对单条时间序列配置不同的参数，也是很难通过少量的人力配置完所有参数的。退一步讲，就算通过人力配置好了告警参数，随着时间的迁移，业务曲线的走势也会发生变化，以前配置的告警策略有可能无法自动适应现在的环境，又需要投入巨大的人力去重新配置告警参数。

【解决方案】

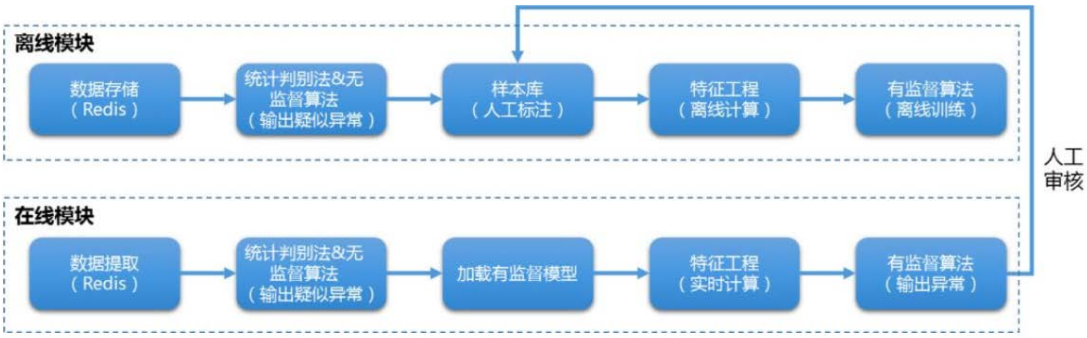

图 1　时间序列异常检测的技术框架

上图为时间序列异常检测的技术框架，作为时间序列的异常检测模型，整体框架分成三大板块，第一个是离线训练模块，第二个是在线预测模块，第三个是 AB test 调优模块。

1) 离线模块，使用统计判别和无监督算法输出疑似异常，例如使用 3-Sigma 原理，Isolation Forest 等算法。然后把输出的疑似异常给人工进行审核，然后加入正负样本库。然后通过提取时间序列的特征，加入有监督算法进行离线训练并且输出模型；这里的有监督学习算法可以使用线性回归，逻辑回归，决策树，随机森林等算法。
2) 在线模块，使用加载离线训练好的模型，并且使用相应的有监督学习算法进行实时预测，也就是判断正常还是异常。在这里，也需要加入人工校正的过程，把误告和漏告的样本加入样本库；其中的 AB test 模块是作为调优的工具，一旦有某个流量的模型效果好，就会全网发布，实时预测。

注：统计判别和无监督算法可以使用 3-Sigma 原理，Isolation Forest 等算法。有监督学习算法可以使用线性回归，逻辑回归，决策树，随机森林等算法。

3、总结

针对海量时间序列异常检测的问题，我们构建了基于机器学习的海量时间序列异常检测方案。该方案把整个过程划分成了无监督，有监督，人工决策三部分。通过运维人员的业务经验，使用机器学习来主动学习人工经验，来实现时间序列异常检测的智能化。

案例 2：金融场景下的根源告警分析

作者：沈建林 @京东金融基础开发部 负责人，技术专家

1、案例概述

金融场景下对线上故障排查的时间要求非常严苛，核心业务要求在分钟级能找到问题的原因，而应用数目和服务器数目又非常庞大，以京东金融为例，单个应用的实例数就有上千之多，应用的数量也是有几千个。如此大的规模下，靠人工经验去排查问题很难达到时效性要求，所以京东金融智能运维平台引入了更智能化的方法来进行根源告警分析。

2、根源告警分析处理流程

根源告警分析是基于网络拓扑，结合调用链，通过时间相关性、权重、机器学习等算法，将告警进行分类筛选，快速找到告警根源的一种方式。它能从大量的告警中找到问题的根源，因此大大缩短了故障排查及恢复时间。

处理步骤：

1) 告警过滤(将告警中不重要的告警以及重复告警过滤掉)
2) 生成派生告警(根源关联关系生成各类派生告警)
3) 告警关联(同一个时间窗内,不同类型派生告警是否存在关联)
4) 权重计算(根据预先设置的各类告警的权重，计算成为根源告警的可能性)
5) 生成根源告警(将权重最大的派生告警标记为根源告警)
6) 根源告警合并(若多类告警计算出的根源告警相同，则将其合并)
7) 根据历史告警处理知识库，找到类似根源告警的处理方案，智能地给出解决方案。

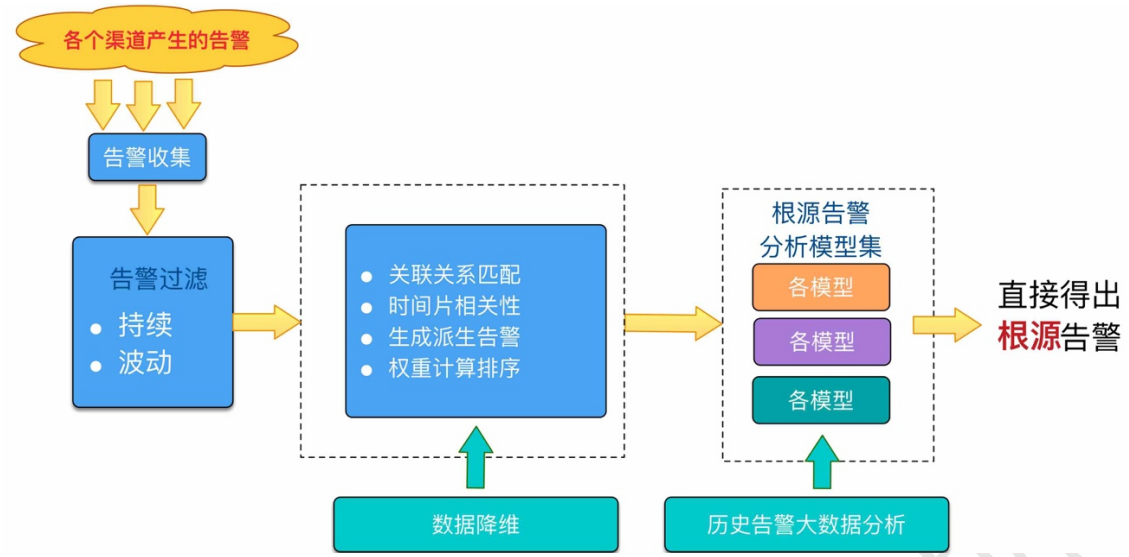

图 1 京东金融根源告警架构图

举例来说：

假设多个系统通过 RPC 进行服务调用，调用关系如下：

D 系统-> C 系统-> B 系统-> A 系统

当 A 系统查询数据库出现查询超时后，告警会层层往前推进，导致 B、C、D 系统均有 N 个超时告警产生。此时，ROOT 分析可以将告警进行收敛，直接分析出根源告警为 A 系统访问数据库异常，导致 A、B、C、D 多个系统异常。

这样，就避免了处理人员和每个系统开发人员沟通，辅助处理人员快速定位问题根源、提高了平均解决时间(MTTR)。如下图所示：

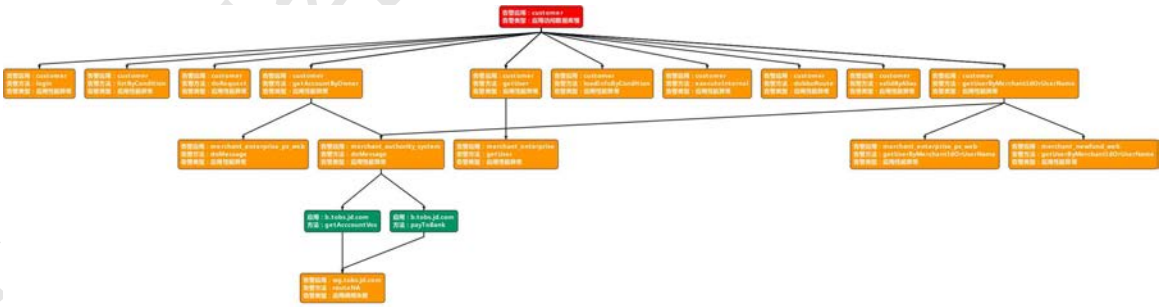

图 2 京东金融根源告警调用链关系图

根源告警 普通告警 无告警　2018-04-12 12:23:02 至 2018-04-12 12:23:46　查看拓扑图

应用/主机	事件	告警时间	服务方法	告警类型
customer	db_slow-mysql-slow(764937939)	12:23:02		应用访问数据库慢
customer	AVG-com.mysql.jdbc.PreparedStatement.executeInternal	12:23:39	executeInternal	应用性能异常
customer	AVG-com.chinabank.cmcs.api.controller.ServiceController.doRequest	12:23:25	doRequest	应用性能异常
customer	AVG-$Proxy78.listByCondition	12:23:22	listByCondition	应用性能异常
customer	TP99-com.chinabank.cmcs.api.user.UserApi.loadInfoByCondition	12:23:20	loadInfoByCondition	应用性能异常
customer	AVG-com.chinabank.cmcs.api.user.UserApi.getUser	12:23:28	getUser	应用性能异常
merchant_enterprise	TP99-com.chinabank.cmcs.api.user.UserApi.getUser	12:23:46	getUser	应用性能异常
customer	AVG-com.chinabank.cmcs.api.user.UserApi.getUserByMerchantIdOrUserName	12:23:43	getUserByMerchantIdOrUserName	应用性能异常
merchant_enterprise_ps_web	TP90-com.chinabank.cmcs.api.user.UserApi.getUserByMerchantIdOrUserName	12:23:23	getUserByMerchantIdOrUserName	应用性能异常
merchant_authority_system	AVG-com.jd.jr.merchant.api.MasApi.doMessage	12:23:40	doMessage	应用性能异常
b.tobs.jd.com		aN:aN:aN	getAcccountVos	
wg.tobs.jd.com	FAILCODE-com.jd.entserver.gateway.web....GateWayGenericController.routeNA:205	12:23:45	routeNA	应用调用失败
b.tobs.jd.com		aN:aN:aN	payToBank	
wg.tobs.jd.com	FAILCODE-com.jd.entserver.gateway.web....GateWayGenericController.routeNA:205	12:23:45	routeNA	应用调用失败
login.tobs.jd.com		aN:aN:aN	secretkeyCheckAndGetUserinfo	
wg.tobs.jd.com	FAILCODE-com.jd.entserver.gateway.web....GateWayGenericController.routeNA:205	12:23:45	routeNA	应用调用失败
merchant_newfund_web	TP99-com.chinabank.cmcs.api.user.UserApi.getUserByMerchantIdOrUserName	12:23:45	getUserByMerchantIdOrUserName	应用性能异常
customer	TP99-com.jd.jr.merchant.api.DubboQueryRouteApi.dubboRoute	12:23:40	dubboRoute	应用性能异常
customer	TP99-com.chinabank.cmcs.api.account.AccountApi.getAccountByOwner	12:23:24	getAccountByOwner	应用性能异常
merchant_authority_system	AVG-com.jd.jr.merchant.api.MasApi.doMessage	12:23:40	doMessage	应用性能异常
b.tobs.jd.com		aN:aN:aN	getAcccountVos	
wg.tobs.jd.com	FAILCODE-com.jd.entserver.gateway.web....GateWayGenericController.routeNA:205	12:23:45	routeNA	应用调用失败
b.tobs.jd.com		aN:aN:aN	payToBank	
wg.tobs.jd.com	FAILCODE-com.jd.entserver.gateway.web....GateWayGenericController.routeNA:205	12:23:45	routeNA	应用调用失败
login.tobs.jd.com		aN:aN:aN	secretkeyCheckAndGetUserinfo	
wg.tobs.jd.com	FAILCODE-com.jd.entserver.gateway.web....GateWayGenericController.routeNA:205	12:23:45	routeNA	应用调用失败
merchant_enterprise_ps_web	AVG-com.jd.jr.merchant.api.MasApi.doMessage	12:23:29	doMessage	应用性能异常
customer	TP99-com.chinabank.cmcs.api.account.AccountApi.validByAlias	12:23:42	validByAlias	应用性能异常
customer	AVG-com.chinabank.cmcs.api.user.UserApi.login	12:23:39	login	应用性能异常

图 3 京东金融根源告警明细图

3、根源告警分析处理方法

根源告警分析的方法主要分为强关联分析与机器学习两类。

1)强关联数据分析

强关联指的是已知确定的关联关系。如：

- 应用之间的调用链关系
- 数据库与应用服务器
- 网络设备与网络设备、网络设备与应用服务器
- 宿主机与虚拟机关系等

若在同一个时间窗内，有多个强关联的设备或应用服务器同时告警，则大概率认为告警之间存在关联关系。

在权重算法中，有一个重要的规则，链路上存在连续的告警可能存在关联，越靠后的应用越可能是根源。现在我们根据例子，分别计算各类根源告警。

继续使用上面的例子，D 应用->C 应用->B 应用->A 应用->数据库 的异常的情况。

- 首先是计算数据库根源告警。根据数据库关联关系，会派生数据库类型的数据库告警、A 应用告警。还会派生一条应用类型的 A 应用数据库异常告警。

根据数据库派生告警以及数据库与应用的关联关系及权重，可以得出数据库异常导致 A 应用查询超时。

• 接下来是计算应用根源告警。根据调用关系，我们先计算出连续多个应用告警的链路。当前 D->C->B->A 四个应用都有派生告警，满足此规则。

• 然后，找到最靠后的告警应用，也就是 A 应用。列举时间窗口内所有 A 应用的派生告警(可能存在多种派生告警，根据权重计算根源)，将权重最高的派生告警标记为根源告警。

比如：A 系统内部有 2 种类型派生告警，分别是数据库告警、GC 告警。

根据权重计算规则，数据库告警为 90，GC 告警 10，也就是说数据库异常告警权重最高。这时由于数据库根源告警和调用链根源告警一致，会将两种类型的告警合并。最后得出结论：数据库异常导致 A、B、C、D 系统告警。

2)机器学习根源分析

强关联数据分析是对已知告警的关联关系，直接进行根源告警分析。但是有些时候，关联关系是未知的，这时就需要通过机器学习算法，找到告警事件之间的隐含联系，再进行根源告警进行故障预测。

目前，京东金融智能运维平台中使用到的主要是以下两大类机器学习算法。

1、关联规则算法

关联规则算法主要进行了 Apriori 算法和 FPGrowth 两类算法的实践。这两类功能相似，都可以发现频繁项集。经过实测，FPGrowth 比 Apriori 更高效一些。

我们按一定的时间间隔划分时间窗，计算每个时间窗内，各种告警一起出现的频率，找出各类告警之间的关联，最终可按分析出的关联关系，生成根源告警。

关联规则算法的优点在于理解和实现起来比较简单。缺点是效率比较低，灵活度也不够高。

2、神经网络算法

循环神经网络(简称 RNN)是一个和时间序列有关系的神经网络，对单张图片而言，像素信息是静止的，而对于一段话而言，里面的词的组成是有先后的，而且通常情况下，后续的词和前面的词有顺序关联。

这时候，卷积神经网络通常很难处理这种时序关联信息，而 RNN 却能有效地进行处理。

随着时间间隔的增大，RNN对于后面时间的节点相比前面时间节点的感知力将下降。解决这个问题需要用到 LongShort Term 网络(简称 LSTM)，它通过刻意的设计来避免长期依赖问题。LSTM 在实践中默认可以记住长期的信息，而不需要付出很大代价。

对于某类故障引起的大量告警之间，存在着时间相关性。将历史派生告警作为输入，将根源告警类型作为输出。通过 LSTM 提取派生告警特征，建立告警相关性分析模型。这样就可以实时将符合特征的派生告警，划分到同一类根源告警中，帮助用户快速定位问题。

需要说明的是金融本身的业务特点决定了对第三方存在依赖性，因此告警本身的随机性较大，客观上导致学习样本的质量不高，需要长期的积累和修正才能达到比较好的效果，因此对于根源告警，如果有条件取到强关联关系，建议使用强关联分析，能达到事半功倍的效果。

4、总结

针对金融场景下业务规模大，应用关系复杂，依赖层次多，排查问题困难的问题，我们建设了具有智能化的根源告警分析的方案。通过告警过滤，生成派生告警，告警关联，权重计算，生成根源告警，根源告警合并，关联历史告警处理知识库的方案进行处理，通过强关联分析和机器学习的分析方法，找到根源告警，并智能地给出解决方案。

案例 3：单机房故障自愈压缩

作者：曲显平 @百度智能云事业部 技术经理

哈晶晶 @百度智能云事业部 故障自愈方向技术专家

1、案例概述

在大型互联网公司中，单机房故障因为其故障时间长、影响范围大，一直是互联网公司运维人员的心头之痛。在传统的运维方式中，由于故障感知判断、流量调度决策的复杂性，通常需要人工止损，但人工处理的时效性会影响服务的恢复速度，同时人的不可靠性也可能导致问题扩大。

为了解决这类问题，我们针对百度内外部网络环境建设了基于智能流量调度的单机房故障自愈能力。结合外网运营商链路监测、内网链路质量监测与业务指标监控构建了全方位故障发现能力，基于百度统一前端(BFE)与百度名字服务(BNS)实现了智能流量调度与自动止损能力。同时，基于实时容量与实时流量调度自动止损策略与管控风险，实现了任意单机房故障时业务均可快速自愈的效果。当前此解决方案已覆盖百度众多核心业务的单机房故障自愈场景。

2、单机房故障止损流程

一个完整的故障处理生命周期包括感知、止损决策、止损、定位和根因分析、解决问题五个阶段。单机房故障自愈主要覆盖从感知到止损阶段，其中感知阶段依赖监控系统的故障发现能力，止损阶段依赖流量调度系统的调度能力。我们来具体看下百度的监控系统与流量调度系统是如何在单机房故障止损场景中起作用。

1. 故障发现：百度监控平台

百度监控平台，针对单机房止损过程中的可用性场景，覆盖故障发现、止损决策、问题定位各阶段的监控。针对单机房止损依赖的容量管理场景，提供资源类监控采集，为容量规划、扩缩容提供数据支持。数据采集覆盖了从运营商外网链路、百度内部网络设备/链路、服务/实例、机器/容器的全方位数据。智能异常检测、趋势预测、多维度分析、关联分析、服务和链路拓扑分析，实现故障的精准发现和定位。

2. 故障止损：百度流量调度平台

针对百度的网络架构和业务架构（参考图 11-4），我们将流量调度拆分为三层：接入层、服务层、依赖层。

1） 接入层：从外网用户发起请求经过运营商网络到百度统一前端（BFE）使用 DNS 实现外网流量调度。

2） 服务层：从 BFE 流量转发至内网服务使用 BFE 提供的 GSLB 动态负载均衡进行流量调度。

3） 依赖层：内网上下游业务之间的流量转发使用百度名字服务（BNS）进行流量调度。

对于单机房止损场景来说，DNS 流量调度的生效时间较服务层、依赖层的流量调度生效时间要慢很多，所以我们期望在发生某个业务的局部单机房故障时，优先进行服务层、依赖层调度。提升止损时效性。

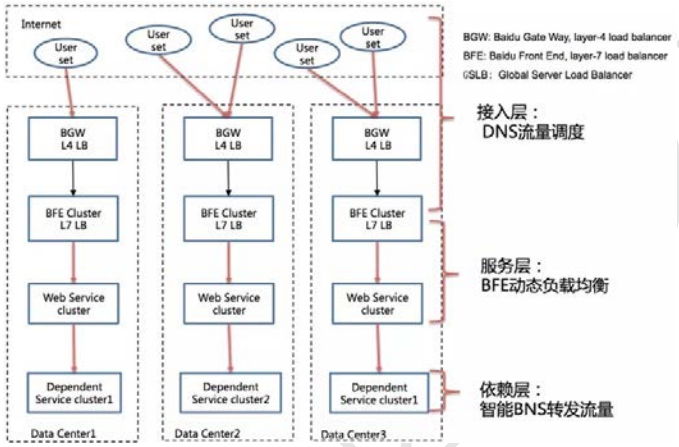

图 1 百度流量接入架构

3、单机房故障自愈的常见问题和解决方案

3.1 容量风险控制能力不足

【问题描述】

传统流量调度的模式有两种：固定比例模式与容量保护模式，参考(图 11.3-2)。

固定比例模式：按照预先设定的固定预案，一个机房故障，该机房的流量按照预先设定的比例分配到其他的机房。很可能某个机房的容量或剩余机房的总容量不足，切流量后导致多个机房发生故障。

容量保护模式：流量调度执行前判断健康机房整体剩余容量是否充足，容量充足则进行流量调度，否则不进行调度并通知人工介入处理。但此种方案面对的问题是：1. 容量 buffer 没有尽可能的利用去实施故障止损。2.容量数据本身存在一定误差，流量成分的变化以及变更等导致的容量退化都可能导致原先容量无法完全可信，依然有流量调度场景下容量过载风险。

【解决方案】

基于容量水位的动态均衡：在流量调度时，对于容量不准确存在的风险，我们划分两条容量警戒线。参考（图 11.3-3）。

安全水位线：流量处于在安全线以下则风险较小，可以一步进行切换。水位上限：该水位线表明服务的最大承载能力，一旦流量超过故障水位线，很大概率会导致容量过载。如果安全水位线提供的容量不足以满足止损，那我们期望使用上两条中间的容量 buffer，同时流量调度过程中进行分步试探，避免一次性调度压垮服务。

基于快速熔断的过载保护：在流量调度时，建立快速的熔断机制作为防止服务过载的最后屏障。一旦出现过载风险，则快速停止流量调度，降低次生故障发生的概率。

基于降级功能的过载保护：在流量调度前，如果已经出现对应机房的容量过载情况，则动态联动对应机房的降级功能，实现故障的恢复。

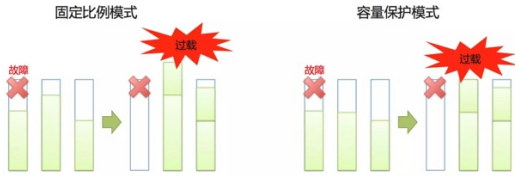

图 2 容量过载案例

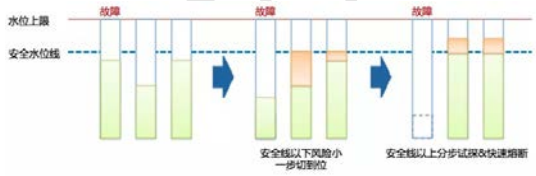

图 3 基于容量水位的动态均衡

3.2 业务线止损策略需求差异大

【问题描述】

我们实现了基础的单机房故障流量调度止损算法，但在部分业务线中仍存在较大的需求差异，比如：

1. 分步动态调度需求：业务存在充 Cache 过程中服务能力降低，需要控制切换速度。
2. 优先级调度需求：产品对延迟敏感，止损时需要优先切到同地域机房；业务服务于多个上游，多个上游的重要程度不同，优先保证重要上游服务稳定。
3. 容量负载计算需求：请求成分不同，不同成分请求带来的容量负载不同。这部分需求一部分与业务强相关，不具备通用性，另一部分则存在不同产品线需求冲突的情况。

【解决方案】

针对以上问题，我们推出了故障止损流量调度策略开放框架。支持用户根据业务需求自定义策略实现。同时将较为通用的策略开放为插件，使业务线可以根据需求自由插拔策略。

策略需求	策略实现
分步动态调度策略	自定义分步调度策略
优先级调度策略	同地域优先调度、业务优先级调度
分成份流量负载计算	分成份流量负载计算策略

图 4 策略分析

基于以上两点，结合智能运维开发框架，单机房故障自愈框架无缝支持不同业务线，使得不同业务的运维人员可以更关注策略本身，而无需关注不同业务线运维模型、底层平台适配成本。

4、单机房故障自愈的架构

百度 AIOps 框架中，单机房故障自愈解决方案构建在运维知识库、运维开发框架、运维策略框架三个核心能力之上（参考图 2）。具体过程为自愈程序搜集分散的运维对象状态数据，自动感知异常后进行决策，得出基于动态编排规划的止损操作，并通过标准化运维操作接口执行。该解决方案策略和架构解耦，并且托管到高可用的自动化运维平台之上，实现了业务在任意单个机房故障情况下皆可自愈的效果。截至目前该方案已覆盖百度大多数核心产品，止损效率较人工处理提升 60%以上。

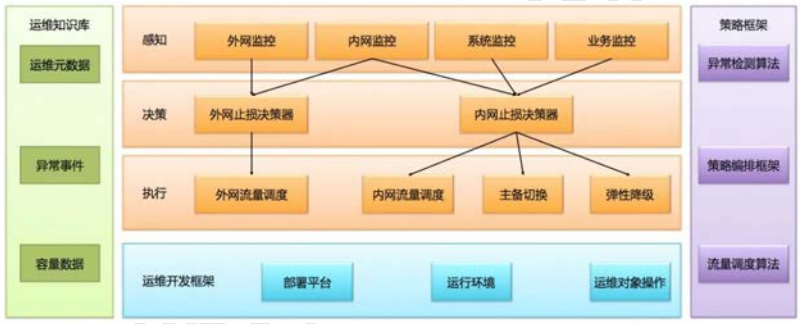

图 5 百度单机房故障自愈系统架构

针对百度服务接入层、服务层、依赖层中监控感知、止损决策与故障止损方式的不同，将止损自动决策拆分为外网止损自动决策与内网止损自动决策。

1. 外网止损自动决策：覆盖接入层。基于外网、内网监控信号；触发外网止损决策器进行止损决策；执行 DNS 流量调度止损。

2. 内网止损自动决策：覆盖服务层、依赖层。基于内网监控、基础监控、业务监控提供的故障信号；触发内网止损决策器进行止损决策；执行流量调度、主备切换、弹性降级等止损操作。

基于 AIOps 的单机房故障自愈方案和传统的故障止损方案对比可参考下表。(表 11-1)

	发现	决策	执行
传统方案	传统监控	人工决策	分散脚本

	1. 固定阈值检测 2. 误报漏报多	1. 局部信息，依赖人工经验 2. 决策速度慢，甚至错误决策，故障扩散	1. 代码质量差，可用性设计欠缺 2. 关键时刻不可用
单机房故障自愈方案	智能监控 1. 智能异常检测算法，动态阈值生成 2. 高召回率、准确率	智能化主动决策 1. 收集全局信息，基于算法自动决策 2. 精准的快速决策，反馈控制	标准执行框架 1. 统一框架和运维托管 2. 开发效率高，高可用执行架构

表 1 传统方案和新方案对比

5、总结

针对传统单机房故障止损方案中存在的问题，我们构建了基于 AIOps 的单机房故障自愈方案。该方案将止损过程划分为统一的感知、决策、执行三个阶段，通过运维知识库解决基础数据、基础设施差异化问题，通过策略框架支持智能化异常检测、策略编排、流量调度问题，同时支持用户自定义策略需求，实现单机房故障自愈的智能化。