基于开源软件自主开发自动化运维系统实践

一、背景与痛点

随着我行业务的迅猛发
系统数量不断攀升，对应的
应的运行与维护工作的难度
维得不到有效施展，越来越
时带来各种运维风险，影响
方面：

1、信息资源数据难管

（1）运用多张 EXCEL
统的软硬件资源信息，信息无法共享和及时同步更新，数据错误率高，容易造成运维工作误判。

（2）为保证 EXCEL 表格数据准确性，需经常进行信息资产盘点，耗时耗力。

（3）数据间的关联无法得到体现，数据用不活，更无法被信息系统利用。

2、基础监控盲点多，覆盖面不全，信息系统故障得不到及时响应，告警信息无法正确匹配软硬件资源而产生误告的痛点。

（1）信息资产多，信息更新快，监控部署和清除跟不上变化，需经常查漏补缺，未及时被监控的系统风险极大。

（2）未被自动化运维系统管控的计算实例，无法自动采集信息、批量查询、操作和巡检等，需要人工梳理和发现。

（3）未被监控和自动化运维的计算实例无法通过有效地手段告知运维人员。

3、运维人员忙乱于繁杂的软硬件与运行环境的部署、安装、创建与配置，整体运维效率低下，精细化水平不高。

（1）运维人员的大部分精力耗费在运维环境部署上，尤其体现在新系统上线频度高，上线量和环境部署量大时，因时间不足，得不到更高运维价值的锻炼，运维眼界和思路得不到开拓。

（2）单个系统逐个手动部署、安装、创建和配置，容易遗漏、错配，或者配置需保持一致的多台计算实例产生差异化。

（3）监控、备份等运维服务得不到重视，运维人员在大量完成应用环境准备后，无精力去部署监控和备份，造成系统上线后无有效监控的问题。

4、存在直接运维操作风险，运维人员水平参差不齐，无法调动更多的运维人力参与运维工作，释放更多操作员的主观能动性，团体

运维的价值和力量体现严重欠缺。

（1）运维人员少，工作压力大，操作员多，但工作范畴相对单一，但介于运维技术的专业性，无法将操作员更好融入运维团队。

（2）人工巡检直接登陆系统，通过超级用户巡检，风险隐患极大，巡检人员素质参差不齐，需要靠运维人员实时跟踪其巡检，耗费精力，其他事情搁置。

（3）每次人工巡检的结果未归档和保留，更无法查询，造成运维巡检数据丢失，得不到有效积累，更无法通过历史运维数据挖掘深层次信息。

5、日常运维巡检过程可能因巡检误操作带来衍生风险，需巡检的软硬件设备与日俱增，巡检效率低下，遗检漏检等现象严重。

（1）巡检点多，类别多，涵盖面广，依靠人工单个巡检无法面面俱到。

（2）人工巡检直接登陆系统，通过超级用户巡检，风险隐患极大，巡检人员素质参差不齐，需要靠运维人员实时跟踪其巡检，耗费精力，其他事情搁置。

（3）每次人工巡检的结果未归档和保留，更无法查询，造成运维巡检数据丢失，得不到有效积累，更无法通过历史运维数据挖掘深层次信息。

6、应用的资源和环境申请源源不断，导致了运维人员大量时间花费在环境部署和复核方面，未及时复核的、不满足配置与基线规范的系统往往存在较大的系统、数据库、中间件、高可用等安全风险隐患，随着系统运行和业务激增，业务故障后，方才发现配置的不合规性。

（1）基础架构的标准规范无法得到有效落地，形成一纸空文。

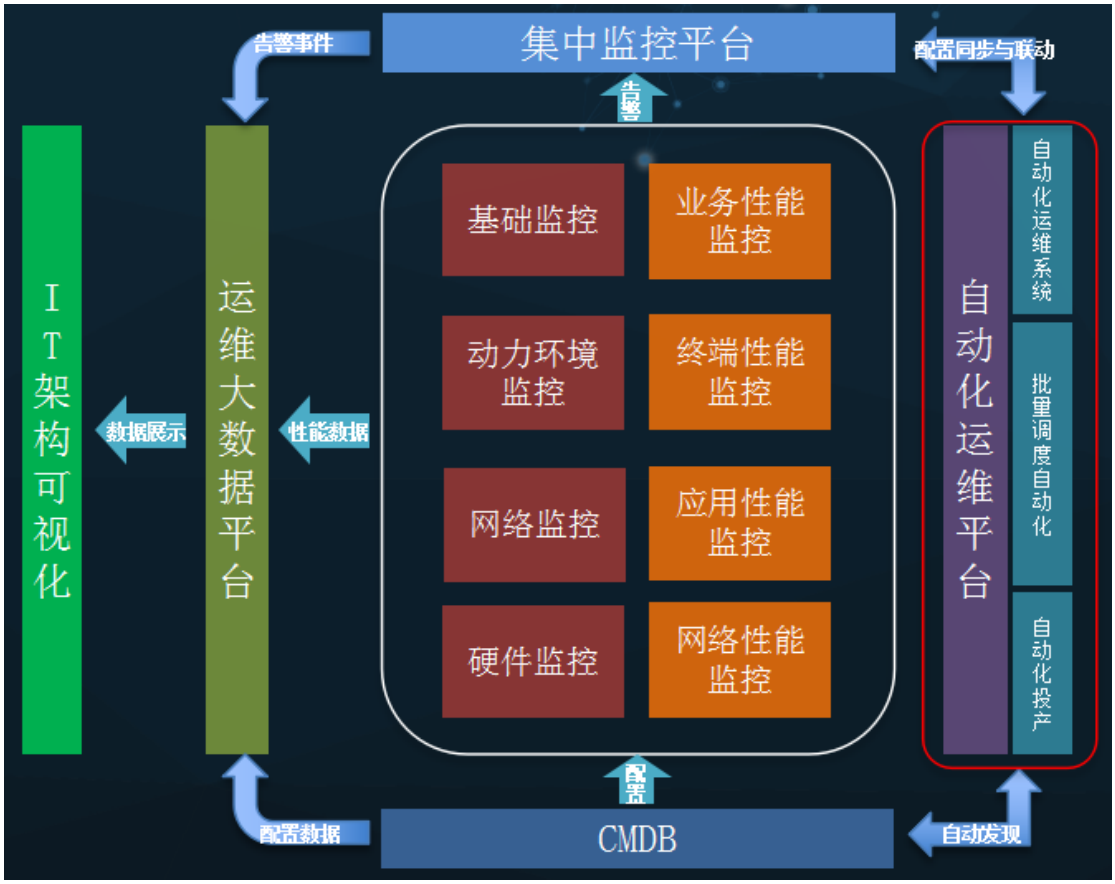
（2）系统上线前基础环境的配置与基线误配、错配和漏配导致的运行风险陡增。

（3）依靠人工的检查和上线前基础环境复核的点过多，难以面面俱到，同时复核到位程度也跟运维人员的能力有关，隐藏配置不合规风险问题依旧存在。

二、总体规划

为有效解决我行现有痛点，对照我行固本强基、提质增效的工作总要求，我科坚决革故鼎新，在运维领域坚持自主运维与科技创新齐进，推动运维领域工作迈向信息化、数字化、自动化、智能化、场景化转型。因此在我行现有基础监控、动力环境监控、网络监控、硬件监控、业务性能监控等监控子系统，和集中监控平台、运维大数据平台、运维流程平台等统一平台的基础之上，进一步拓展、扩大运维体系架构和覆盖范围，例如终端性能监控、应用性能监控和网络性能监

控等监控子系统，统一 CMDB 平台、自动化运维平台和 IT 可视化平台等统一平台。整体规划架构图如下所示：



1、监控体系架构简介：满足业务系统端到端监控的需求。建立用户 APP、WEB、客户端等终端的终端性能和体验监控系统，通过不同的方式收集各类终端的行为或体验数据，接入后端运维大数据，为不同的用户的行为进行画像，供精准营销或者风控项目消费，进一步指导业务的运营和管理；建立从业务层、网络层和应用层三个层面和角度的专业监控系统，输出各个层面的监控和统计指标，满足故障排查和定位根因提供不同角度的监控支持，而非传统监控的只发现现象而无法定位根源点和原因；结合现有各项基础监控子系统，全面实时掌控业务系统各个层面的指标状态，及时定位故障根因，提升业务系统连续性。

2、自动化运维体系架构简介：搭建自动化运维系统、自动化批量调度、自动化投产上线三个维度的自动化体系，结合上层可集成整合化的自动化运维平台，满足生产系统端到端自动化运维的需求。加速端到端运维交付的质量和规范性，减轻运维工作成本，释放运维动能。

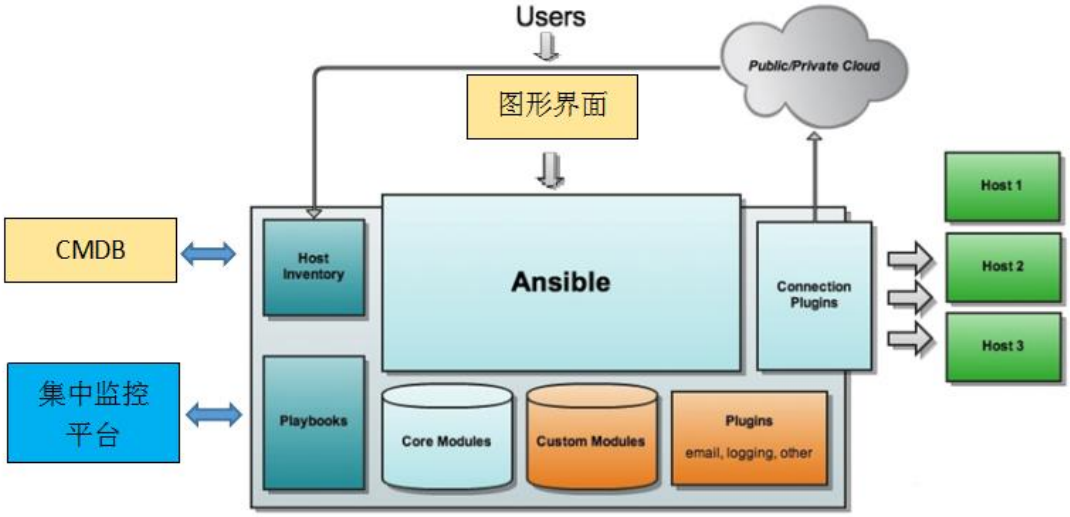
3、智能运维体系架构简介：智能运维需要数据才能有产出，基础配置数据来源于 CMDB，分析数据来源于各不同角度的监控子系统，通过建立运维大数据平台，来整合所有的基础性能数据、用户终端性

能数据、网络性能数据、业务性能数据和应用程序性能数据等指标型数据，事件告警数据、应用日志数据和系统日志数据等日志型数据，甚至网络报文数据等，产出例如告警事件与指标性数据的关联，进行智能分析，得出可能的原因，定位告警源。运维大数据不仅仅是简单的数据集中化和展示，更深层次的目标是多数据源的挖掘和分析，需结合人工智能技术深入探索运维智能化的落地与见效。

4、多系统、平台间联动体系简介：统一 CMDB 为所有系统和平台提供统一的配置基准数据，提升联动的数据质量和效果；自动化运维平台自动采集和发现价值数据和数据关联，供其他系统和平台使用，和各项资源建立自动化关联关系,提供不同自动化运维场景调用 API，供其他系统和平台调用；集中监控平台对接所有监控系统和平台，实时收集所有事件和告警，结合 CMDB 配置数据，第一时间匹配和丰富事件告警内容,以丰富的通知手段和详尽真实的告警详情告知相关负责人；运维大数据通过多样化、不同通道的方式，集成各系统和平台的实时或历史的结构化、非结构化数据，并进行过滤、清洗、加工、整合、分析、输出和数据持久化；IT 架构可视化系统通过业务系统部署架构图、业务逻辑架构图、业务网络拓扑图三类架构图的方式，结合运维大数据中，不同数据源的数据，包括智能运维产出的建议，进行实时的展示，让数据和图联动，更为直观的展示业务系统整体运行状况。运维以 IT 架构可视化为主，智能运维为辅，强调人在运维中不可替代性。

三、自动化运维系统实践

在以上的总体规划基础之上，我科全面展开了自动化运维系统、批量调度自动化、自动化投产三位一体的自动化运维平台建设工作，通过以上项目的建设,可以很好的满足端到端运维自动化的工作任务。由于整体文章篇幅的问题，笔者下面将重点介绍其中之一的，基于开源 ansible 软件和 cmdbuild 软件，自主部署的自动化运维系统。我科通过 Shell 脚本，成功开发了若干实用功能的自动化、批量运维的友好窗口界面，并自主搭建了 CMDB（配置管理数据库），便于软硬件资源集中管控。该系统大幅提升了运维工作的效率，进一步减轻了运维人员的工作压力，并标准规范化了运维操作，同时规避了人工直接运维带来的操作风险;该系统涵盖了生产交易类与管理类系统近 2000 余个计算实例的批量运维工作。自动化运维系统整体架构如下图所示：



- 用户通过窗口界面调用 Ansible，来向受管控的主机执行各种命令，实现各类功能场景，包括监控批量部署、备份批量部署、日常批量运维、软件批量安装、批量巡检、配置基线批量核查等等；
- Ansible 所在服务器定期执行任务计划，自动收集受管控主机的各类软硬件配置信息，并同步至 CMDB 中；
- Ansible 所在服务器定期执行任务计划，将 CMDB 中的软硬件配置信息同步至集中监控平台，监控平台的告警事件的信息更精准；
- Ansible 所在服务器定期执行任务计划，将集中监控平台中的监控点与 CMDB 中的软硬件配置信息进行比对，发现尚未部署的监控点，将该信息上送给 CMDB，便于运维人员查看；
- Ansible 所在服务器定期执行任务计划，同步 CMDB 中需要自动化运维的主机信息，并检测是否都已具备自动化运维条件（互信、Python 安装等），并将尚不能进行自动化运维的主机信息上送给 CMDB，便于运维人员查看；
- Ansible 所在服务器定期执行任务计划，比对集中监控平台中的监控分组信息与 CMDB 中软硬件配置信息，发现尚未纳入监控分组中的主机，将该信息上送给 CMDB，便于运维人员查看；
- Ansible 所在服务器定期执行任务计划，将一体化流程平台中的软硬件资源申请相关信息同步至 CMDB 中，减轻人工录入的工作量。

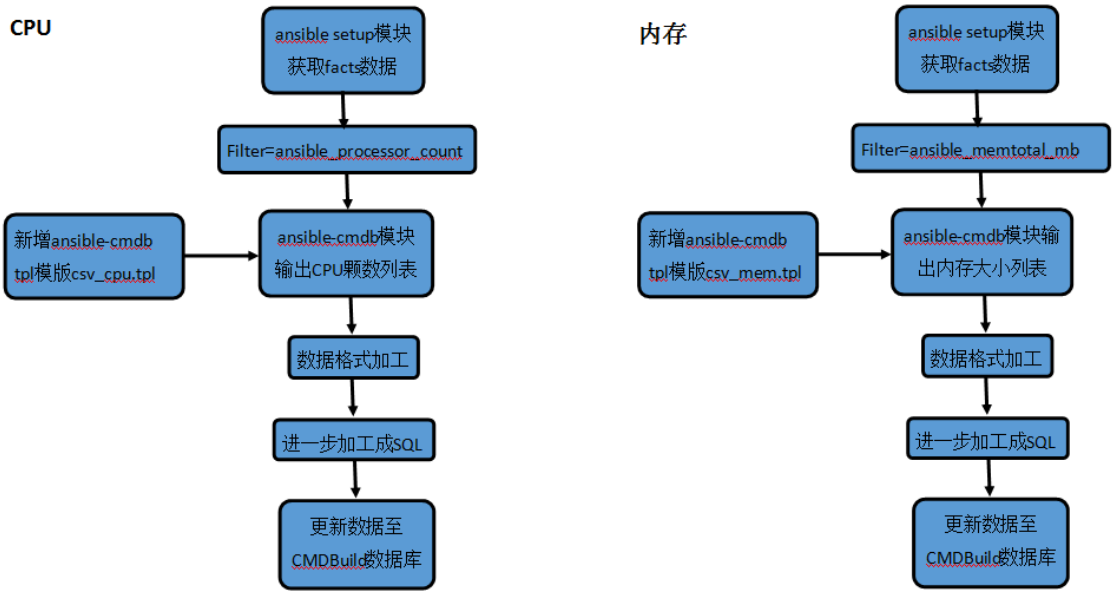
下面对其主要功能、实践方案和实践效果详细说明。

1、理顺双数据中心软硬件资源及关联关系，将数据纳入自主开发的信息系统管理（CMDB），结合自动化运维系统对计算实例中软硬件资源及软件版本的自动化采集和对接将来的 RFID 射频定位数据，自动化地精准软硬件资源的信息数据。消除目前的信息资源数据难管理、难使用、准确性低的痛点。

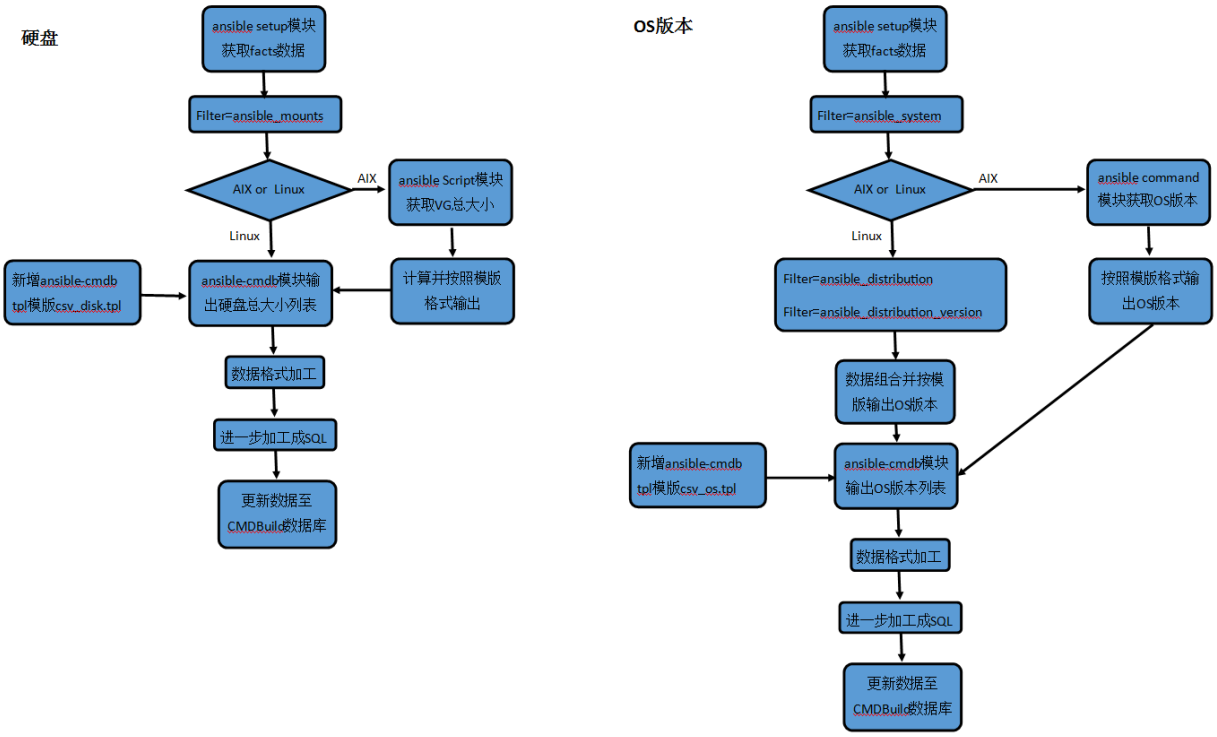
- 实践方案：

（1）CMDB 与 Ansible 环境搭建：通过开源 CMDBuild 搭建 CMDB 配置管理系统，建立自上而下的数据表和表间关联关系，导入和梳理现有软硬件资源及配置数据；Ansible 环境搭建可参考笔者另一篇文章：Ansible 自动化运维体系在生产环境下实践（11 步极快速搭建）

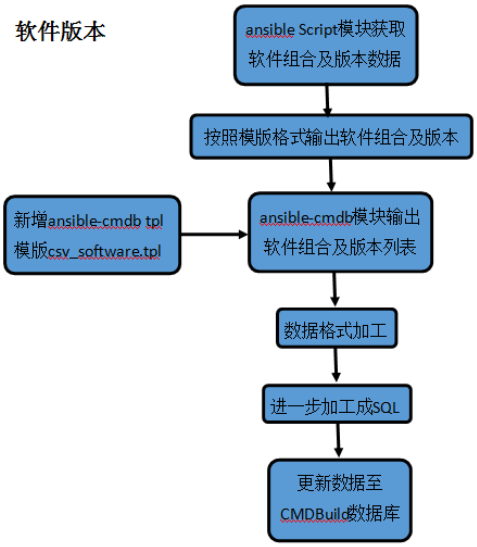
（2）CPU 和内存大小自动获取：通过 Ansible setup 模块获取各主机 facts 数据，通过 Filter 过滤出 CPU 颗数和内存总大小；编辑 Ansible-cmdb 的 TPL 模板,利用 Ansible-cmdb 模块输出主机 IP、CPU 颗数与内存大小列表，并对该列表的格式进行加工、转换成适合进一步加工成 SQL 语句的 csv 格式文件，最终运行 SQL 语句更新 CPU 和内存数据至对应的 CMDB 表中。方案逻辑流程如下图所示：



（3）硬盘容量、操作系统版本自动获取：通过 Ansible setup 模块获取各主机 facts 数据，通过 Filter 过滤出各硬盘容量大小和操作系统类型。编辑 Ansible-cmdb 的 TPL 模板，通过对操作系统类型进行判断（AIX 或者 Linux），当为 AIX 操作系统时，调用 Ansible Script 模块，将脚本注入各主机执行，获取所有 VG 的总大小，或者通过调用 Ansible Command 模块获取 AIX 操作系统详细版本，其后再通过计算并按照模板格式输出；当为 Linux 操作系统时，直接采用 setup 过滤后的值。这些指标最终统一通过 Ansible-cmdb 模块输出为主机 IP、硬盘总容量、操作系统版本列表，并对该列表的格式进行加工、转换成适合进一步加工成 SQL 语句的 csv 格式文件，最终运行 SQL 语句更新硬盘容量、操作系统版本数据至对应的 CMDB 表中。方案逻辑流程如下图所示：

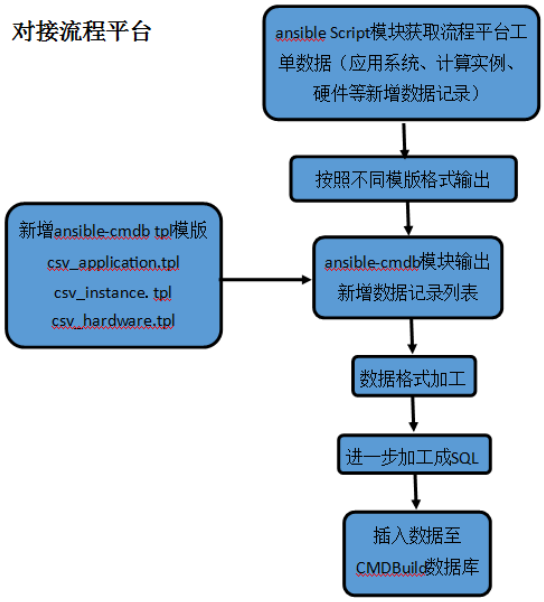


（4）软件组合和版本自动获取：调用 Ansible Script 模块，将脚本注入各主机执行，获取该主机上安装的标准软件和对应版本等数据；编辑 Ansible-cmdb 的 TPL 模板，按照模板格式输出软件组合和版本；通过 Ansible-cmdb 模块输出主机 IP、软件组合和版本信息列表，并对该列表的格式进行加工、转换成适合进一步加工成 SQL 语句的 csv 格式文件，最终运行 SQL 语句更新软件组合和版本数据至对应的 CMDB 数据表。方案逻辑流程如下图所示：



（4）同步流程平台数据：调用 Ansible Script 模块获取流程平台的软硬件资源申请、应用系统上线等工单数据，涉及对新增的应用系统、计算实例和硬件等数据记录的自动同步；编辑 Ansible-cmdb 的 TPL 模板，按照模板格式输出新增数据记录列表，并经过数据格式加

工、转换成适合进一步加工成 SQL 语句的 csv 格式文件，最终运行 SQL 语句插入相关数据至对应的 CMDB 数据表。方案逻辑流程如下图所示：



➢ 实践效果：

（1）搭建 CMDB，将数据信息纳入信息系统中管理，统一数据查询与更新界面，实现数据共享，在同一数据使用界面，统一运维，保证运维的同一数据基础。

（2）无需经常进行数据资产盘点，数据信息通过自动化手段采集+同步资源申请单中的数据+枚举数据来确保数据的准确性。

（3）CMDB 与监控、自动化运维系统自动联动，加上人工经常查询使用，数据用活了，进一步确保了数据的准确性。

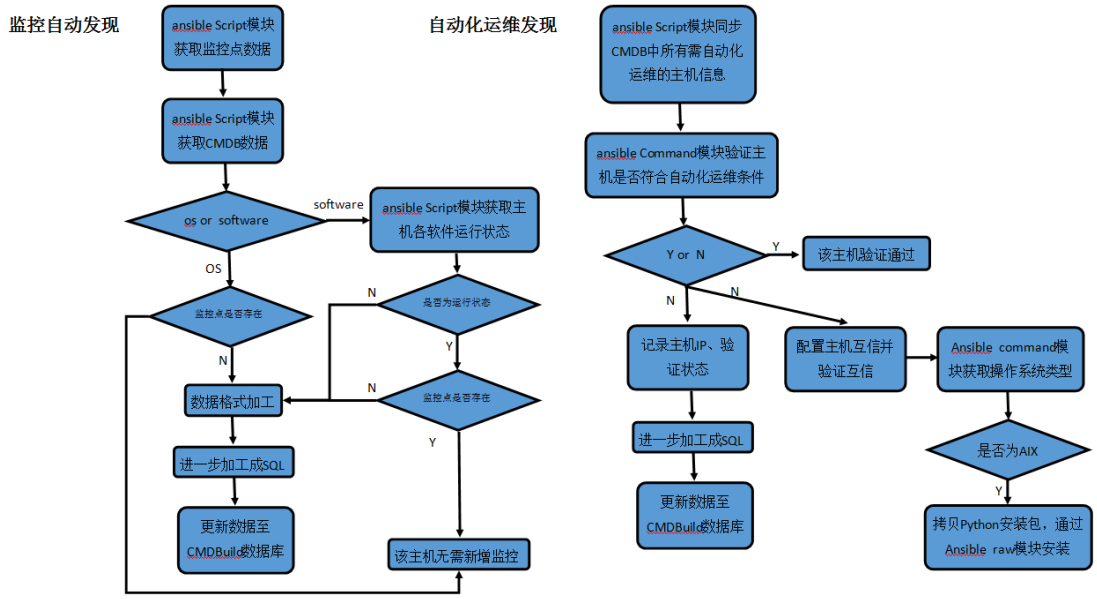
2、自主实现监控点的自动发现、自动化运维节点自动发现，并与 CMDB 实现联动，可视化展现尚未监控和纳入自动化运维体系的计算实例。消除目前基础监控盲点多，覆盖面不全，信息系统故障得不到及时响应，告警信息无法正确匹配软硬件资源而产生误告的痛点。

➢ 实践方案：

（1）监控自动发现：调用 Ansible Script 模块分别获取监控平台所有监控点的数据信息，包括操作系统、数据库、中间件、日志、PING、关键端口等，并抽取 CMDB 数据表的记录；通过判断 CMDB 中的软件组合和操作系统类型，来检测该主机的监控点是否全覆盖，其中 CMDB 中的软件组合信息还需结合各软件的运行状态来判断该监控点是否是真实需要的；最后通过进一步的数据加工、生成 SQL 语句、运行 SQL 语句更新相关数据至对应的 CMDB 数据表，完成监控自动发现功能逻辑。其逻辑流程如下图所示：

（2）自动化运维发现：调用 Ansible Script 模块获取 CMDB 中所

有需自动化运维的主机信息；调用 Ansible Command 模块验证主机是否符合自动化运维条件，例如 SSH 互信、Python 及版本；如果符合，通过验证，如果不符合，并行执行两种逻辑，一是对主机 IP 和验证结果中的状态值进行记录，进一步加工成可运行的 SQL 语句，更新 CMDB 中相应的数据表；二是配置主机互信、验证互信，并通过 Ansible Command 模块获取操作系统类型，如果为 AIX 操作系统，还需额外通过 Ansible Raw 模块安装 Python 环境（标准 Linux 自带 Python）。其逻辑流程如下图所示：



➢ 实践效果：

（1）CMDB 中录入的所有信息资产，均能够及时发现尚未监控的点，自动化的将相关信息列举。

（2）未被自动化运维系统识别的计算实例，能够自动地发现，运维人员可及时将其纳入自动化运维体系。

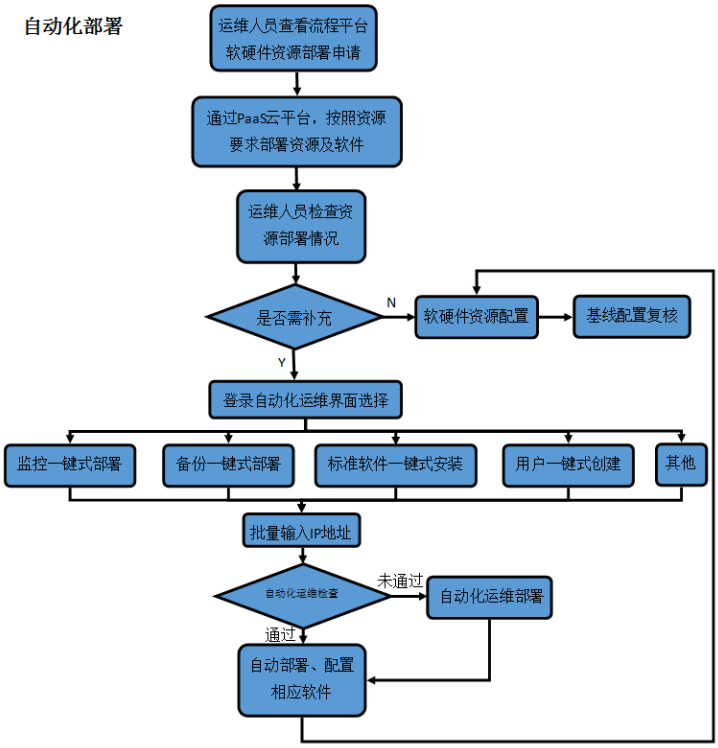
（3）自动发现监控点和自动化运维节点后，直接将信息同步至 CMDB，并产生告警事件，提醒运维人员。

3、自主实现批量自动化运维一键式部署、批量监控一键式部署、批量备份一键式部署、批量用户一键式创建、批量常用软件安装和配置。将运维人员从原来繁杂的部署、安装、创建、配置中解放出来，更高效地完成既定的工作任务，将重心转向更深入、更精细化的运维当中去，运维人员忙而不乱。

➢ 实践方案：

软件及运行环境自动化运维部署是我行 PaaS 云平台的补充，首先由资源申请人员通过流程平台对软硬件资源进行申请，并填写资源需求，架构人员根据资源需求，将需求格式化成可落地的标准信息，经过层层审批之后，最终落实到运维人员实施阶段；通过 PaaS 级云平台，自动编排需部署的资源和软件平台，并检查是否需自动化运维

系统补充安装软件运行环境，如监控、备份、用户、其他标准化的数据库、中间件等软件；如果需要补充，则由运维人员登录自动化运维界面选择相应菜单，并批量输入 IP 地址，后续批量部署工作将由自动化运维系统自动完成，包括自动化运维检查，未通过后的自动化运维部署，和软件及运行环境的安装配置等；其他额外无法通过 PaaS 云平台和自动化运维系统实现的软硬件安装配置工作均手动由运维人员实施；最后由专门基线配置复核人员，通过自动化运维菜单，选择进行相应的基线配置自动化复核。其逻辑流程如下图所示：



➢ 实践效果：

（1）通过菜单式的一键式部署界面，运维人员只需批量输入 IP 地址既可完成部署，释放运维人员压力，减轻工作任务。腾出更多的时间精细化运维工作。

（2）批量部署、自动化安装和配置相关运维环境，配置一致化。

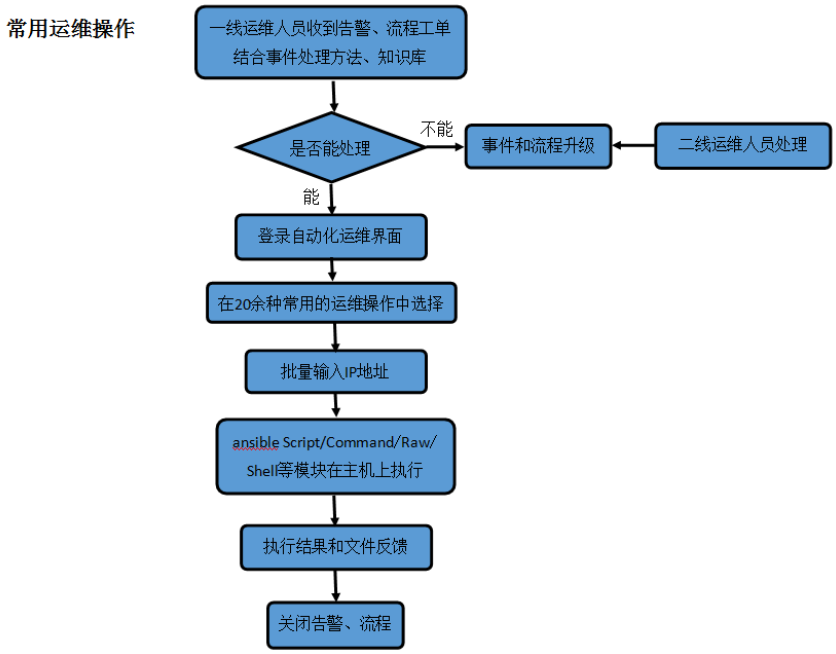
（3）部署仅仅是简单的菜单操作，凡是新上线系统均能及时部署运维环境，未部署运维环境的系统也能及时被发现。

4、将常用运维批量查询及常用批量操作菜单化，简便运维操作，同时通过菜单背后的标准化运维命令，降低直接运维操作风险，调动更多的运维人力参与运维工作，释放更多操作员的主观能动性，体现团体运维的价值和力量。

➢ 实践方案：

我行将运维人员分为一线、二线，一线运维人员在收到告警或者流程工单时，结合告警事件中自动匹配（匹配告警关键字或情景）的

告警处理方法，和知识库记录，判断自身能力是否能够自行处理或者辅助二线人员远程处理，如果不能，通过流程平台对事件进行升级，交由二线运维人员处理；如果能，则登录自动化运维界面，在不直接登录需维护的主机的情况下，通过在界面中选择 20 余种常用运维操作，并批量输入 IP 地址的方式，自动调用 Ansible Script、Command、Raw、Shell 等模块在主机上执行该运维操作，执行的结果和文件直接在界面上反馈，处理完成并确认无误后，可关闭告警事件和流程。其逻辑流程如下图所示：



➢ 实践效果：

（1）通过将简单的运维查询和操作菜单化，交由一线操作员完成，使其融入运维团队，更好地体现个人价值。

（2）批量查询和批量主机操作，简单快捷。

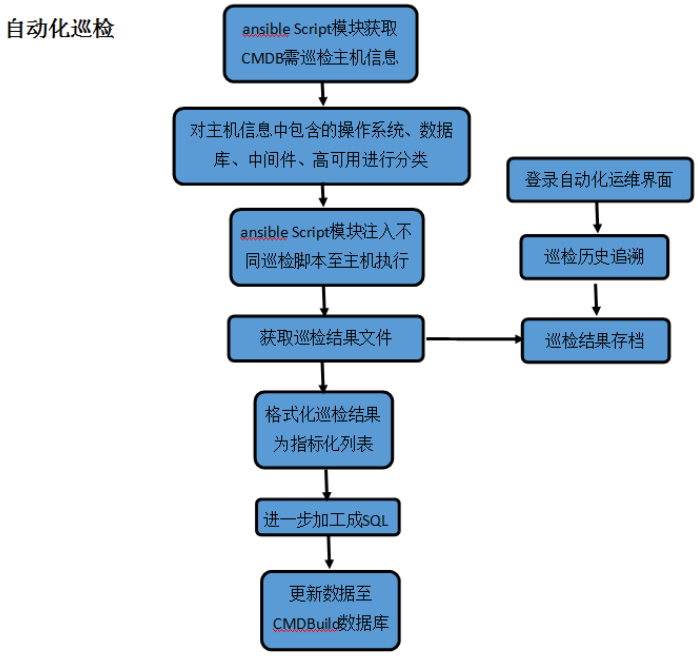
（3）无需登陆的菜单化运维杜绝操作风险。

5、自主实现故障日志一键式收集、运维一键式巡检及巡检报告生成、集中归档与查询功能。巡检人员无需登陆每个系统逐个巡检，通过自动化运维系统的操作菜单，批量输入 IP 地址，即刻开始自动巡检其上可能存在的数据库、中间件、操作系统和高可用架构软件等，并生成巡检报告。巡检报告可通过该系统集中归档和查询。通过该功能降低了可能因直接登陆巡检误操作带来的衍生风险，同时提高了巡检效率，避免了巡检遗漏等现象。

➢ 实践方案：

目前我行需人工巡检的主机数和软硬件运行环境越来越多，已经严重占用了大量维保方和行方运维人员的宝贵时间，无法集中精力运维更有价值的工作任务，因此，我行通过将自动化巡检纳入到自动化

运维系统。首先通过 Ansible Script 模块获取 CMDB 中所有需巡检的主机信息，并结合自动采集到的软件组合和版本、操作系统等进行分类的；其次再通过定时自动调用 Ansible Script 模块，将各类巡检脚本注入至不同批次的主机中执行，获取巡检结果并存档、归档；最后自动将巡检结果进行指标格式化处理，通过进一步加工成可执行的 SQL 语句，插入数据至 CMDB 相应的数据表，运维人员可直接查看该数据表，更好的了解该指标的实际值和标准基线值/范围间的差异；为更精细化的了解巡检结果，专家运维人员可直接登录自动化运维界面，查看、下载、追溯各主机的巡检历史详细结果。其逻辑流程如下图所示：



➢ 实践效果：

（1）自动化地发现其上的各类数据库、中间件、操作系统和高可用架构等，自动化地巡检，无法人工干预，巡检结果面面俱到。

（2）通过运维一键式菜单，和人工批量输入 IP 地址开始巡检，无需人工跟踪，无需逐个通过超级用户登陆系统，规避人为误操作风险。

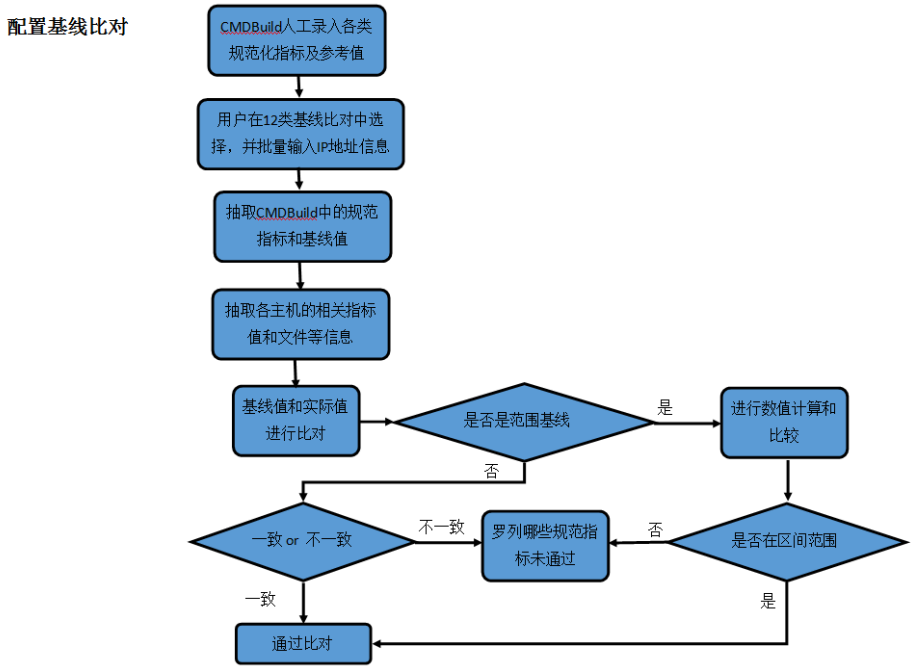
（3）巡检结果可直接下载，供各领域专家查阅，也可直接通过 CMDB 查看。巡检结果集中保存和归档，便于以后调阅，甚至对接运维大数据，供数据挖掘和分析。

6、自主实现系统上线配置与基线自动化、批量核查。应用的资源和环境申请源源不断，导致了运维人员大量时间花费在环境部署和复核方面，未及时复核的、不满足配置与基线规范的系统往往存在较大的系统、数据库、中间件、高可用等安全风险隐患，随着系统运行和业务激增，业务故障后，方才发现配置的不合规性。通过该功能能

够在系统上线前，自动化比对标准规范与实际落地的配置参数间的差异性，即刻进行整改，完成标准的应用环境交付。

➢ 实践方案：

将标准规范进行表数据格式化，人工梳理各类规范化配置参数，如 AIX、Linux 操作系统规范，VIOS 配置规范，HACMP、TSA、GPFS 高可用规范，双活规范，WAS、MQ 中间件规范，DB2、ORACLE、MYSQL 数据库配置规范等等；用户通过自动化运维菜单选择 12 类基线比对中的某项比对功能，并批量输入需比对的 IP 地址信息；自动化运维系统通过 Ansible Script 模块，注入相应脚本至 CMDB 服务器，抽取 CMDB 中的规范指标和对应基线值；通过 Ansible Script 模块，注入相应脚本至各主机，抽取相关指标值和文件等信息；自动化运维系统结合基线值和实际值，根据是否是范围基线，来分类比对，若是范围基线，则进行相应数值计算和比较，若不是范围基线，则直接比对基线值和实际值是否一致；最终判定配置通过基线比对，或者罗列哪些规范指标未通过，及时告知复核人员。



➢ 实践效果：

（1）标准规范由文字转为数字信息，更容易更新和保存，更容易得到自动化运维系统利用，实现落地。

（2）通过自动化运维系统的配置核查，一键式检查出与标准规范不符的地方，将风险扼杀在系统上线前。

（3）标准规范与高水平运维人员的经验相结合，转为数字信息，复核不存在死角，覆盖面广、全、自动化实现。