# 智能化软件开发大脑 – UDev draft4

Table of Contents

[智能化软件开发大脑 – UDev draft3 1](#_Toc529978843)

[背景 1](#_Toc529978844)

[痛点及现存解决方案 2](#_Toc529978845)

[定义 2](#_Toc529978846)

[设计原则 2](#_Toc529978847)

[UDev优势 3](#_Toc529978848)

[UDev劣势 3](#_Toc529978849)

[核心技术（列表） 3](#_Toc529978850)

[技术难点 4](#_Toc529978851)

[UDev体系架构 4](#_Toc529978852)

[案例分析 5](#_Toc529978853)

[物体识别系统（draft1） 5](#_Toc529978854)

[舆情分析系统与垂直搜索引擎（draft1） 6](#_Toc529978855)

[通讯录（draft1） 10](#_Toc529978856)

[5分钟视频介绍（TODO） 10](#_Toc529978857)

[需求 10](#_Toc529978858)

[核心技术剖析 10](#_Toc529978859)

[代码搜索技术 10](#_Toc529978860)

[API推荐技术 10](#_Toc529978861)

[服务服用技术 11](#_Toc529978862)

[代码生成技术 11](#_Toc529978863)

[SDLC Agile Model 12](#_Toc529978864)

[未来 13](#_Toc529978865)

[TODOs 13](#_Toc529978866)

[参考 13](#_Toc529978867)

## 背景

* 提高软件开发的效率和质量是软件工程领域的研究者与实践者始终追求的重要目标。如何在软件开发的过程中，通过改善软件开发方法和运用技术手段提高软件开发的自动化水平，以达到降低软件开发成本、提高软件生产效率、改善软件产品质量的目的，是众多软件研究工作的共同目标。围绕这一目标，软件工程领域的研究者一方面通过不断探索新的软件形态、探究新的环境下的软件本质来创造新的软件开发范型和软件架构；一方面也通过不断借鉴各个科学领域的研究成果，在本领域现有的开发范型和架构下不断改善技术方法提高生产效率。近年来，人工智能技术取得了长足的发展和进步，这一进步也对软件工程领域的研究形成了重要的促进。软件工程领域的研究者在既有的软件开发范型和体系架构上不断将新的智能技术引入现有的方法和技术中，从而形成了大量有价值的智能化软件开发方法和技术成果，这些新的成果不但促进了软件工程本领域的技术和方法的进步和革新，同时也对人工智能及其他相关领域的发展起到了重要的推动作用。

## 痛点及现存解决方案

* 需求频繁变更
  + 敏捷和迭代式开发
* 项目维护漫长且困难
  + 云端，SaaS
* 风险，成本重要指标预测不准
  + 预测大约范围，而非准确值
* 人员流失造成供给方缺陷
  + 更舒适环境，提高工资待遇
* 流程的繁杂，需求说明书，信息传递的损失
  + 删减掉冗余中间环节，让初始需求方直接参与到软件研发全流程中

## 定义

* 一款基于人工智能，群体智慧的软件研发与服务平台，为项目需求方，交互设计师及算法工程师等团队成员提供研发流程优化，代码自动生成等技术。基于敏捷软件开发原则，对从业务初始需求分析到最终成品交付流程进行深度优化。

## 设计原则

* UDev摒弃传统瀑布式软件开发模式，立足于敏捷软件开发，为B端和C端客户提供**第三代颠覆性智能化软件研发新模式**
* 充分吸收全球互联网公司敏捷开发原则：**小步快跑，快速迭代**
* 业务逻辑，人工智能等数据服务**以流水线为核心，切分为若干个小步骤**（如数据抓取，预处理，模型训练，模型预测和参数调节等），在业务流图上优化次序任务，高度并行可并行任务
* 尽早且持续的交付软件迭代原形，使目标客户纳入研发闭环中（参与感提升）
* 在人工智能数据服务中大量使用**预训练模型**，并在其基础上作个性化优化训练，研发时间约节约30%（？）
* 研发过程中欢迎随时改变需求，并把此变更作为经验训练数据供UDev大脑进行训练
* 经常性地交付可以工作的软件，交付间隔缩短到周，交付时间间隔越短越好
* 核心团队耦合度高，高度依赖面对面交谈和信息传递
* 以能工作的软件为核心，非面面俱又死板的文档
* 随时响应需求变更
* 坚持数据驱动
* 强调目标与关键成果（OKR）

## UDev优势

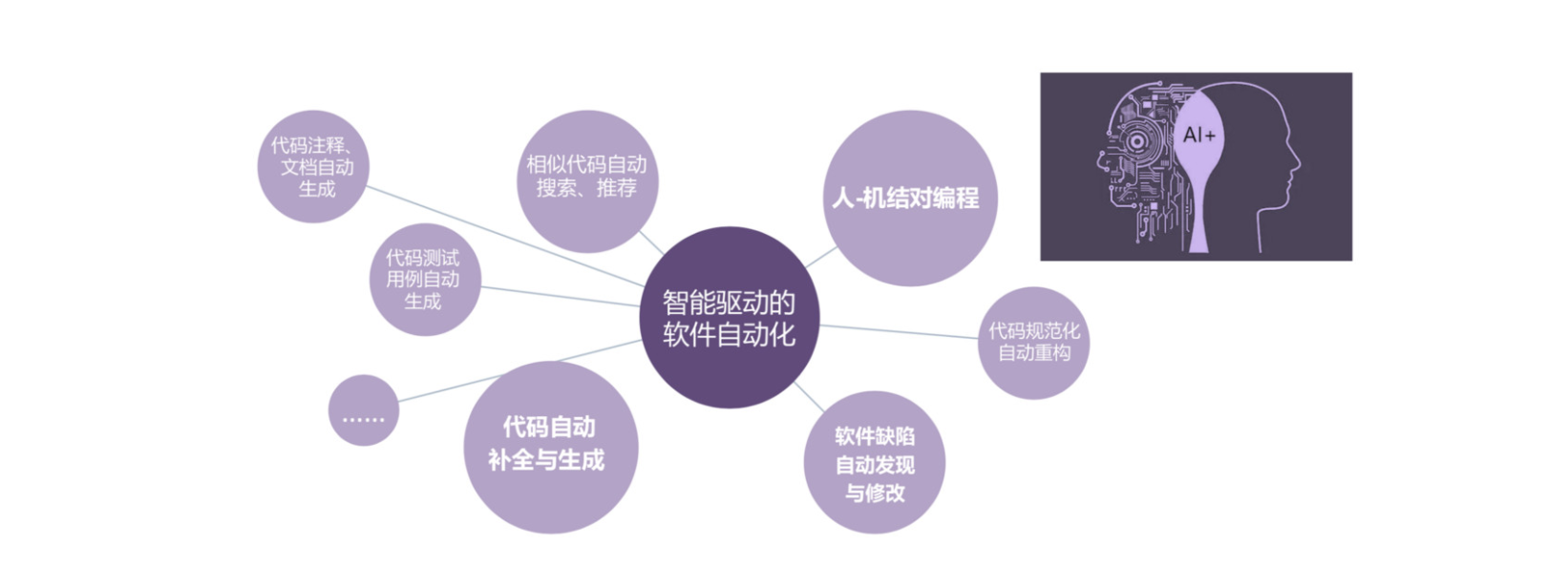
* Promotes teamwork and cross training
* Functionality can be developed rapidly and demonstrated
* Delivers early partial working solutions
* Minimal rules, documentation easily employed
* Enables concurrent development and delivery within an overall planned context
* Gives flexibility to developers

## UDev劣势

* Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction
* There is a very high individual dependency, since there is minimum documentation generated
* Transfer of technology to new team members may be quite challenging due to lack of documentation

## 核心技术（列表）

* 以深度学习作为工具
  + 提供草图到代码生成功能（如pix2code ）
  + 提供机器人和工程师结对编程，以加速软件开发过程
* 业界领先的
  + 代码搜索技术
  + API推荐技术（相似代码智能推荐）
  + 服务复用技术
  + 业界领先的代码生成技术
  + 基于语义的代码自动补全
* 预设丰富技术架构
* 预设丰富模版
* 预集丰富软件库

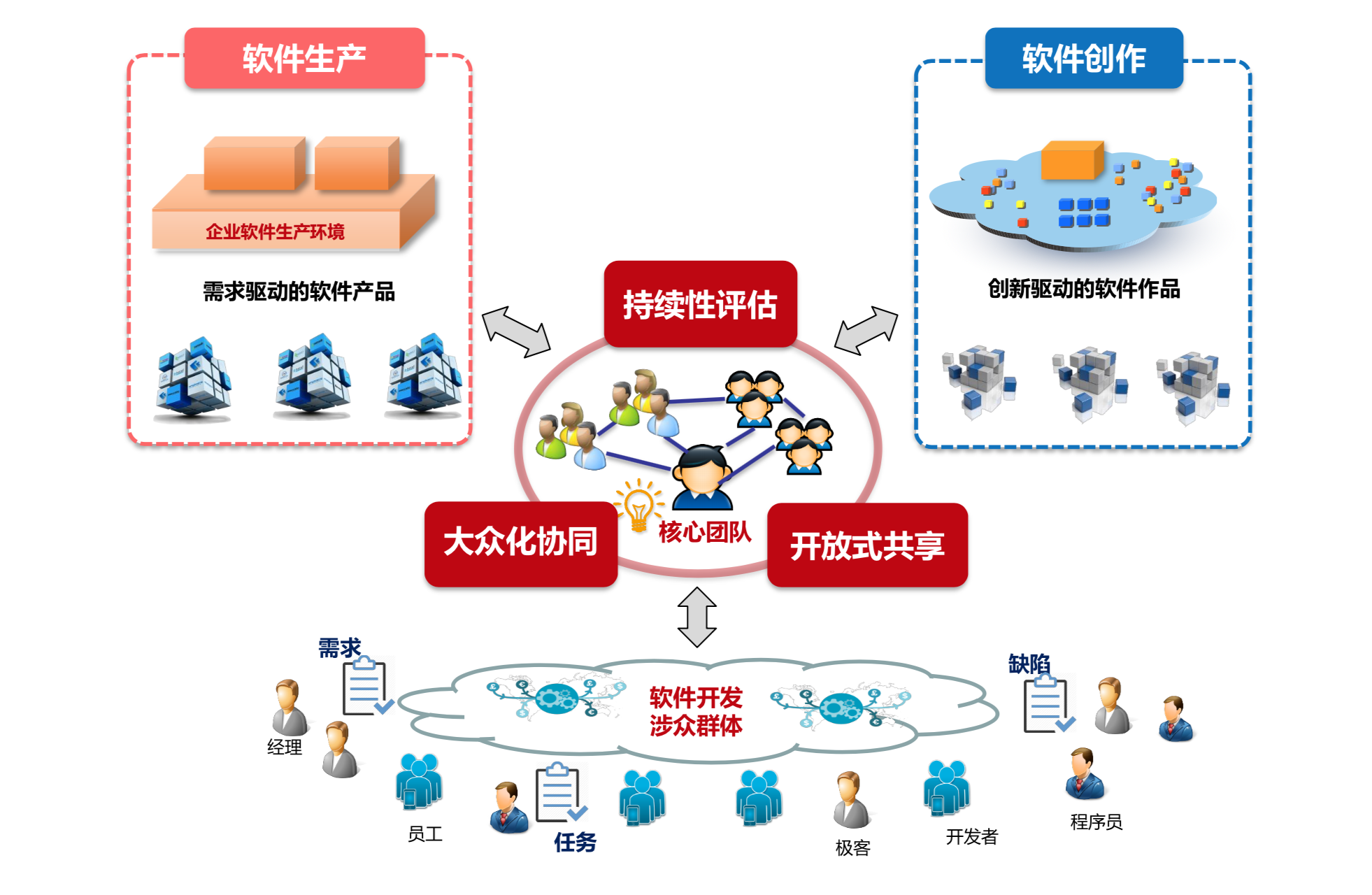


## 技术难点

* 对于数据集的描述和操作（如转换，更新，编辑和删除）
* 用户交互场景的描述（如此用户在手机上会完成哪些操作，涉及到哪些数据集的更新，即前端用户交互部分）
* 如何有效明确客户需求，把信息转化为用户场景，子场景
* 根绝用户场景，定义有限种UI 交互操作
* 大规模编程部件复用
* 使用前沿前端技术，着重可重用性
* 数据源多样性和兼容能力
* 高效交互

## UDev体系架构

* 有明确的核心和外延
* 注重生态圈建设
* 模块化程度高



## 案例分析

### 物体识别系统（draft1）

需求方 & 研发团队

步骤：

* 给出 问题和数据集。在这里，Happy Whale DB中的2,5000图片，做Whale Identification
* 给出 问题评价标准。在这里，每幅图片对应5个标签。这幅图片里面出现了哪条鲸（每条鲸有一个ID编号）
* 定义团队OKR
  + 目标O：构建单条鲸鱼物体识别引擎
  + 关键成果KR：整体准确度需达到80%上（和现有准确度作交叉对比）
* 潜在挑战与风险：对于每一只鲸，只有有限的几张图片

人员构成

* 1产品经理
* 4算法工程师（人工智能，计算机图像方向）

引擎部件与架构体系：

* 数据源与爬虫系统
* 预处理系统
* 深度学习物体识别引擎
  + 注：当准确度（或其他模型衡量指标）在某阀值下时被触发再训练。通常为离线训练，在线预测。
* UI前端交互
  + 预测数据展示，用户审核，AI正负反馈闭环，打标签数据。前端交互是人工智能系统必不可少的组成部分。

### 舆情分析系统与垂直搜索引擎（draft1）

需求及需求分析（来自一个国家层面的真实客户）：

从整体来看，即构建一个垂直搜索引擎

1. the system should able to check simultaneously the different web sites following a key word

* 每个受控网站都需要维持一个关键字黑名单，如此关键字出现，则需要上报
* 有一强大爬虫系统，此系统能够周期性爬取目标网站内容，如发现违规，即上报

1. the global matching and localization of the sites following their IP address

* 维持一个国家层面的数据库，包括IP地址和IP所属地点

1. able to reduce the access or to restrict the access to these web sites

* 类似中国互联网防火长城，能够把一些敏感网站排除在外，并且禁用VPN服务

1. following the activities of different pages in the different social media

* 此需求没有弄明白

1. ability to restrict the access to the pages (shouting) the activities using URL or other tool

* （同 Point 3）

1. checking the activities through VPN or other cyber soft and limit or restrict the flux

* 使用舆情分析系统能部分检测到所谓的cyber soft和flux，但无法控制

1. identify the users and location of the persons behind the different accounts

* 通常使用IP地址来反向追踪其所在位置

1. to secure the activities (virus and other massive attacks)

* 为国家建立一个类态势感知平台/SIEM/SOC

1. the global trend and the different connections of the cyber activists

* （同Point 8）

1. to reach all the information based on the influence and the followers of the different pages in the social network and social media

* 舆情分析系统的一部分功能

1. All other items who could be integrated, and related to the cyber activities, cyber crimes and counter insurgency (Globally, we need the equipment, tools and software and training)

* 舆情分析系统/SIEM/SOC/态势感知

研发团队

步骤：

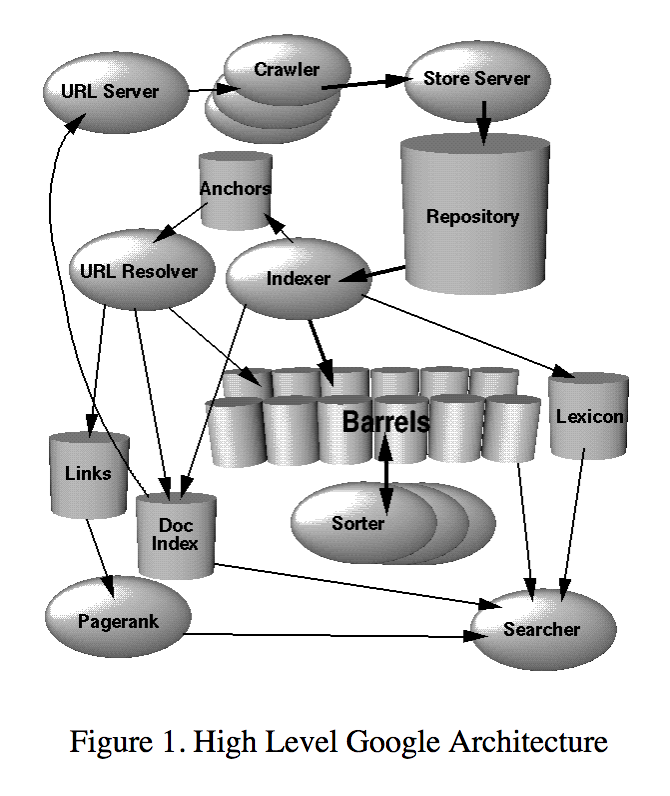
* 给出 问题。通过关键字搜索猜测搜索者意图，准确返回不同社交媒体如Twitter，微博，Google，各大新闻媒体对某一主题/舆情的正负面评价。
* 给出 数据集。数据的收集从最重要的数据源开始收集，对于国内，重要的数据源有新浪微博，微信公众号；对于国外，有twitter，Google等。
* 给出 问题评价标准。对于这个搜索引擎的质量评价，学术上可以使用NDCG，工业上可以收集用户点击，翻页行为等，以作衡量
* 定义团队OKR
  + 目标O：构建一个基于文本的垂直搜索引擎，以作舆情分析之用
  + 关键成果KR：
    - 单个AND查询响应时间维持在ms级别
    - 索引文档数目需要达到亿级别
    - 当网站内容发生变更时，在1天时间内需要在搜索结果中体现
* 潜在挑战和风险：
  + NA

人员构成

* 1产品经理
* 4算法工程师（搜索引擎方向）

引擎部件与架构体系：

* 爬虫Crawler
* 索引Indexer
* 排序Ranker
* 大数据存储与计算
* UI交互



UI整体呈现1

Crawler:

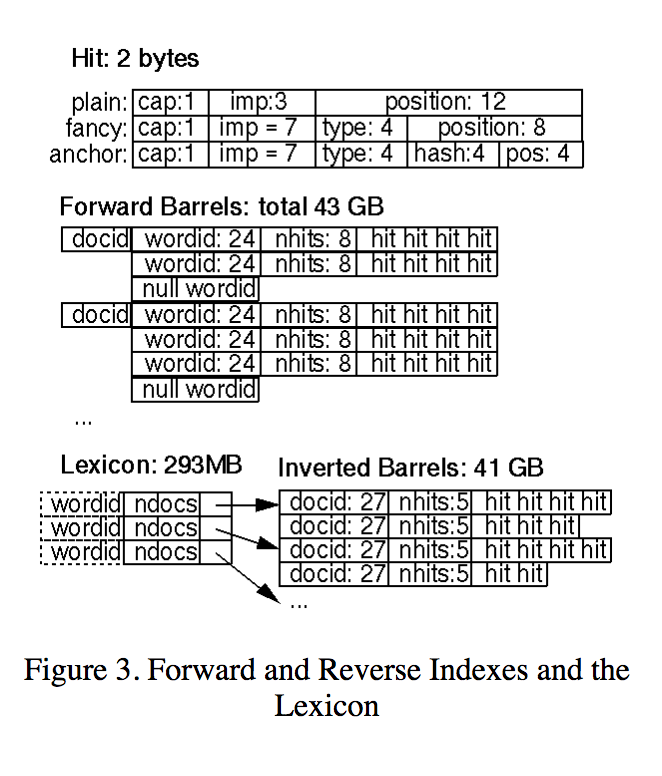
* 输入：URL列表
* 输出：下载好的网页以及分析出来的一批新URL

Indexer：

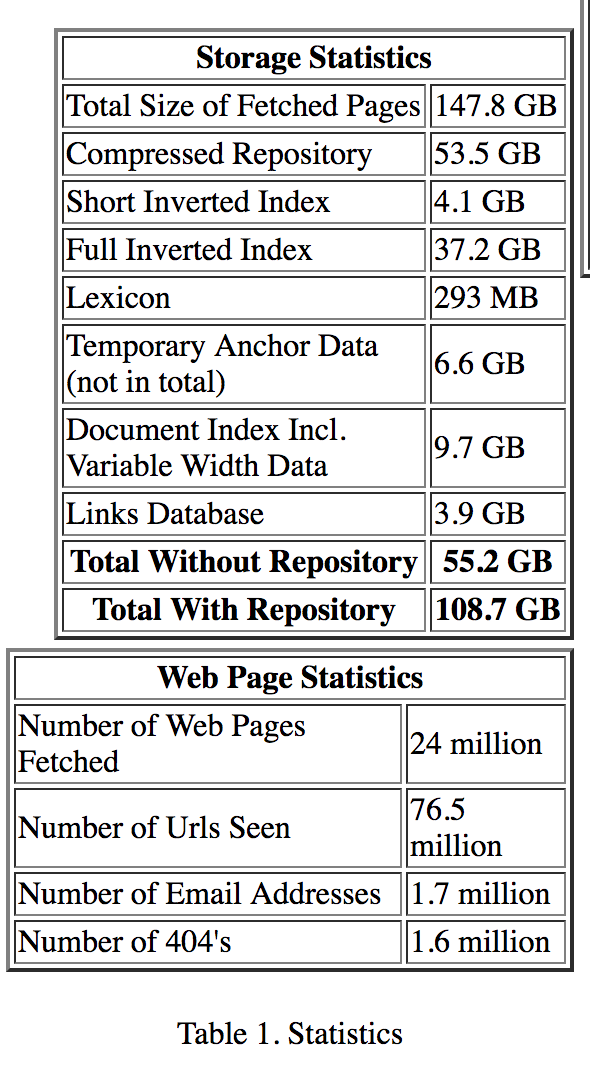
* 输入：原始网页
* 输出：倒排索引和前向索引

Ranker：

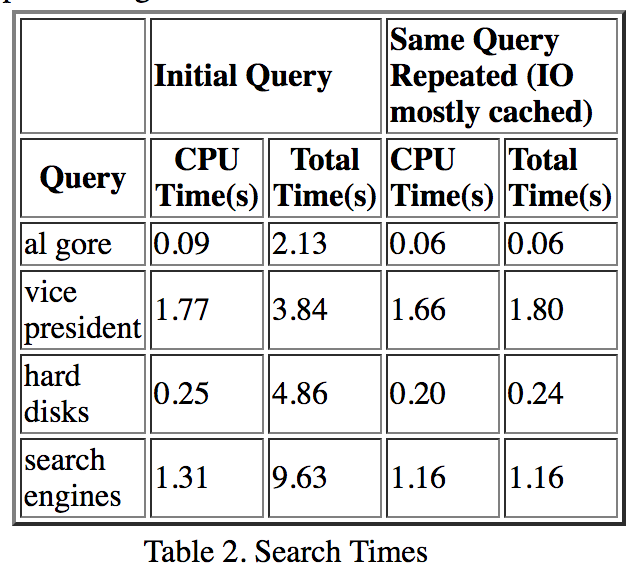
* 输入：用于查询的索引
* 输出：排序后的查询结果，用户隐形和显性反馈



关键数据呈现：Lexicon，前向和倒排索引



关键数据呈现：存储系统



关键数据呈现：在线搜索

### 通讯录（draft1）

* 需求：由计算机自动设计并生成一通讯录APP
* 需求分析：背后的原始数据为(name, value) pair。由核心数据衍生出多种表现形式和多种用户操作，如新增，删除和编辑。这部分可以使用模版预设，提前完成80%的工作。最后需加入用户个性化需求，此是剩余10%的研发工作。

## 5分钟视频介绍（TODO）

### 需求

* 按时顺
* 需场景化
* 突出多数客户经统计分析后的共性
* 突出客户的个性化需求分析
* 体现适应能力强且快的特点
* 有可演示的视频

## 核心技术剖析

代码搜索技术

工业界和学术界一直以来都比较关注代码搜索的研究，Singer等人通过研究指出代码搜索是开发者最频繁的行为之一。随着数以万计的开源代码项目变得可用，代码搜索在软件开发中的作用变得更加突出。一项针对谷歌开发者的调查显示，在谷歌的代码搜索记录中，一个开发者平均每个工作日要进行12次代码搜索查询。在所有的查询中，期望获取代码片段的查询比例占到了34%。近年来，基于网络的在线搜索网站纷纷提供代码搜索的服务，包括Google (通用搜索平台)，Krugle，Google Code Search，GitHub和SourceForge等，根据用户的输入进行搜索。从2005年开始，在软件工程顶级会议、期刊上发表的代码搜索的研究论文越来越多。按照输入类型来划分，可将代码搜索的研究分为多个子研究方向：自由文本，代码上下文，结构化信息，和其它输入类型。在上述各个类别的检索技术中，智能检索技术不断被引入进来，并与一些其他相关信息相结合，如代码之间的调用关系，用户对代码段的反馈信息，这些智能检索技术的使用，进一步提高了代码检索结果的质量。

API推荐技术

为了提高软件复用的封装性、降低复用难度，很多可复用软件都是通过API（Application Programming Interface，应用编程接口）的形式对外提供服务和功能访问的。早期的API的主要是各种程序库，并且随着开发语言及各种软件开发社区的发展，这类API的数量急剧上升，例如JDK1.8中的基础类库就提供了超过14万个方法。近年来，随着各种开发框架和基础软件（例如安卓系统）的发展，许多框架和系统基础功能和服务也通过API的方式提供。例如，Eclipse的图形编辑框架GEF（Graphical Editing Framework）提供了6万多个API方法，而安卓开发框架则提供了5万多个API方法。随着互联网和云计算的发展，越来越多的网络化服务平台也都选择通过开放API的方式提供平台服务。随着API数量的增多，开发人员在选择和使用API方面的代价也越来越大。开发人员首先需要根据开发任务的需要找到合适的API，然后需要获取API的使用说明和样例代码等参考信息，同时还要在遇到异常和错误等问题时获得问题解决方案和指导。针对这些问题，研究人员开展了一系列研究工作。早期研究工作主要关注于利用程序分析和信息检索等技术手段，辅助开发人员搜索API及相关信息；此后，随着互联网推荐技术的发展，一些研究者开始研究基于上下文等信息进行API的推荐。为了支持开发人员正确使用API，一些推荐方法还特别关注于API使用模式的挖掘和推荐；近两年，随着深度学习等智能化技术的发展，一些面向API推荐和问题解决的智能化开发技术也逐渐涌现，关注于相关知识的抽取和复用。

服务服用技术

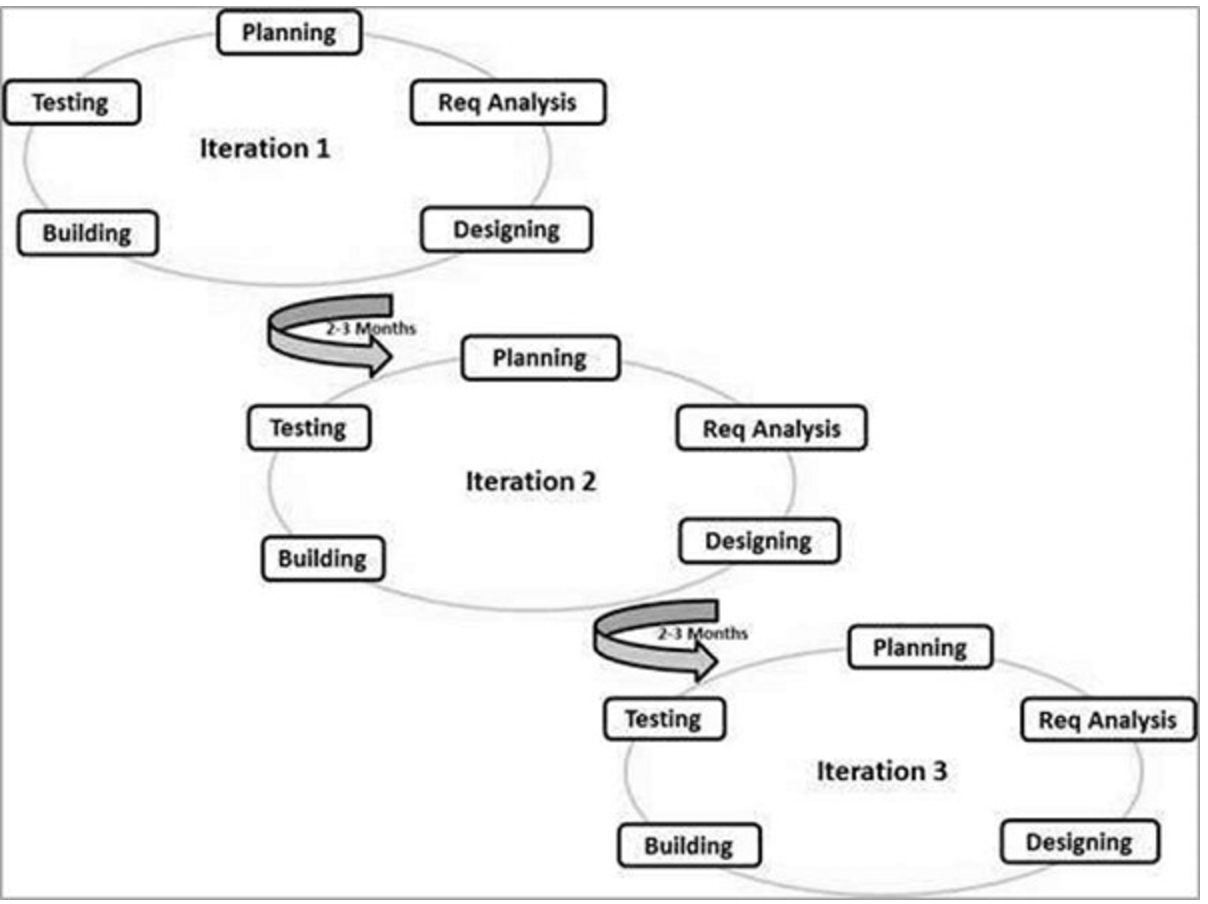
随着互联网技术的发展和云计算平台的逐渐成熟，服务化软件（Service-Based Software）成为一种重要的软件形态。这类系统以Web服务为基本构成元素，以服务组装为核心构造方法。在Web服务形态方面，主要体现为技术标准之争，经历了从SOAP、WSDL、WS-\*技术为代表的学院路线，到HTTP、JSON、RESTful API为代表的产业技术路线的发展历程。而智能化研发技术，主要体现在服务组合技术方面。服务组合作为面向服务计算的核心研究点之一,能利用服务的重用价值,满足用户的复杂请求,实现服务增值。通过智能化服务组合技术可以灵活、高效地实现业务流程构造，快捷地完成新业务。围绕服务组合建模语言、智能化服务组合方法、服务组合执行与验证等方面进行研究，国内外学者启动了诸多研究项目，如METEOR-S、SEFL-SERV、e-Flow等。智能化Web服务组合问题已经成为分布式计算、流程管理以及软件工程领域的一个研究热点。目前，工业界和学术界已提出了众多的Web服务组合方法，从Web服务组合实现方式角度可以分为服务编制和服务编排，根据组合的动态性程度可以分为静态服务组合和动态服务组合，根据组合的自动化程度可以分为手动服务组合、半自动服务组合和全自动服务组合方法。早期的研究工作主要从业务流程的角度出发，关注静态、手动或者半自动化的服务组合。由于服务的分布、动态特性、用户需求的不确定性以及运行环境的持续演化特性，后期的研究工作主要关注动态的、自动化的服务组合方法。

代码生成技术

计算机自动生成代码是近年来软件工程的研究热点之一。代码自动生成极大的减少了程序员的工作量，提高了开发效率。随着开源社区的发展，我们可以通过分析大量的代码从而进行代码生成。代码自动生成的一大困难在于源代码本身具有诸多的约束和限制。近年来，在原有的基于组合优化方法进行程序综合研究的基础上，产生了一些新的基于机器学习技术进行程序生成的方法。虽然计算机直接生成代码是十分困难的，但是一些简单的代码生成应用可以从一定程度上辅助程序员编程。**代码补全**是代码自动生成的一种较成熟的典型应用，绝大多数的程序员在进行软件开发的过程中会使用框架或库API来复用代码。代码补全机制已经成为现代集成开发环境中的不可或缺的组成部分。在工业界中，不同编程语言的集成开发环境包括：Eclipse, IntelliJ IDEA, Visual Studio和PyCharm都可以支持在程序员输入代码时尝试补全程序的剩余部分。在学术界，1988年研究者就提出用智能化的方法帮助开发人员开发和维护软件。传统方法往往利用代码的静态特征进行代码生成，2012年，Hindle A等人提出利用自然语言的方法来解决程序生成问题，越来越多的研究者将**人工智能的方法用于代码自动生成**。按照所采用技术及应用场景的不同，可将当前的程序生成方法分为两类：一类为基于程序输入输出结果的程序生成，一类为基于程序代码语言特性的代码生成。基于输入输出结果的程序综合主要基于机器学习模型，利用程序输入输出结果之间的对应关系构造训练数据集，并利用该数据集对机器学习模型进行训练，**以达到在输入输出效果上模拟程序行为的目的**。该类方法尤以基于深度神经网络的方法为代表。基于程序设计语言模型的程序生成主要利用程序设计语言自身所具有的统计特性，通过对已有大量程序代码的学习建立相应程序设计语言的机器学习模型，并基于该模型在已有程序代码的基础上通过自动补全的方式生成新的代码。

### SDLC Agile Model

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks. Every iteration involves cross functional teams working simultaneously on various areas like – Planning, Requirement Analysis, Design, Coding, Unit Testing and Acceptance Testing.



Agile is based on the **adaptive software development methods**, whereas the traditional SDLC models like the waterfall model is based on a predictive approach. Predictive teams in the traditional SDLC models usually work with detailed planning and have a complete forecast of the exact tasks and features to be delivered in the next few months or during the product life cycle.

Predictive methods entirely depend on the **requirement analysis and planning** done in the beginning of cycle. Any changes to be incorporated go through a strict change control management and prioritization.

Agile uses an **adaptive approach** where there is no detailed planning and there is clarity on future tasks only in respect of what features need to be developed. There is feature driven development and the team adapts to the changing product requirements dynamically. The product is tested very frequently, through the release iterations, minimizing the risk of any major failures in future.

**Customer Interaction is the backbone of this Agile methodology, and open communication with minimum documentation are the typical features of Agile development environment.** The agile teams work in close collaboration with each other and are most often located in the same geographical location.

## 未来

* 生态圈的建立和完善 = 平台承托 + 企业内核心团队 + 外部开发者 + 用户
* 智能化软件开发是一个正处于高速发展的研究领域，仍有诸多学术研究工作正在推进过程中。除软件工程研究领域外，更有诸多其他研究领域的学者也在关注智能化软件开发相关的研究问题，例如多项程序自动生成相关的研究成果即来自于人工智能领域的研究者。可见，关于智能化软件开发或软件开发自动化问题的研究正在逐步演化为多研究领域交叉的新问题，可以合理预见，在未来的研究中，将形成软件工程与人工智能等相关领域互相推动的局面，智能化软件开发的研究也将逐步成为多领域共同关注的重要问题。

## TODOs

* 在垂直搜索引擎（舆情分析）业务线上开始收集用户潜意识行为和动作
* 支持更多数据源和增强数据兼容性
* 增强UI前端模组化能力
* 增强人才储备：人工智能相关人才

## 参考

（关于“智能化软件工程”）

* aiXcoder 和AI结对编程
* pix2code 从设计到代码生成
* AI Programmer: Using artificial intelligence and genetic algorithms to automatically write programs <https://github.com/primaryobjects/AI-Programmer>
* Neural Programmer Interpreters
* DeepCoder: Learning to Write Programs
* 软件智能化开发技术的进展与趋势 <https://toutiao.io/posts/bgucz1/preview>
* Jinja2 – The Python Template Engine
* Unsupervised Program Induction with Hierarchical Generative Convolutional Neural Networks
* The Anatomy of a Large-Scale Hypertextual Web Search Engine <http://infolab.stanford.edu/~backrub/google.html>