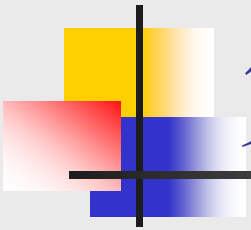
财务决策支持系统

范明



第一部分

信息系统概述

信息系统的定义

- 信息系统是围绕着数据的收集、存贮、处理、传递和使用四个过程的系统。它通过处理数据获得经济活动必不可少的有用知识，提高经济活动的效率和效用。
- 企业运作需要信息系统的支持
- “价格机制”也是一种信息系统。

当代经济条件下，信息系统对企业而言变得必不可少

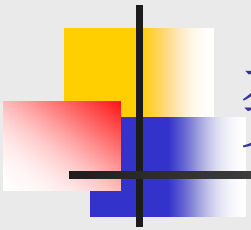
- 经济全球化：管理和控制全球化市场、在世界范围内竞争、组建全球工作组、货物或者产品全球配送系统、经济一体化（INTERNET发展）——企业运作的环境越来越复杂，涉及的因素越来越多，需要记录和处理的信息量越来越大。
- 经济模式变化：知识经济、信息经济、产品周期缩短、竞争越来越激烈、“时机”的重要性越发突出、员工的知识有限性——信息越来越重要，同时高速处理信息的能力也越来越重要和必要。
- 商业企业的变化：传统企业模式和企业行为已经转变，企业扁平化、结构非中心化、制造柔性化——这些企业模式和企业行为要想“玩得转”的充分和必要条件就是：企业具备相当高水平的信息处理能力。

信息技术改变企业的组织形式

- Internet（全球公共网络）：企业可以方便地将组织分布到全世界，组建跨国公司的成本大大降低；资本流动的成本大大降低；雇佣劳动的成本也大大降低。
- Intranet（企业内部网络）：企业内部的协调和通讯变得非常方便，能够进行即时或者快速的决策和反馈；权力非中心化；企业结构扁平化。
- 便携计算：能够快速、低成本地进行复杂决策和预测。
- 图形界面：使用知识和信息的门槛大大降低，知识或者信息应用的“傻瓜”化。

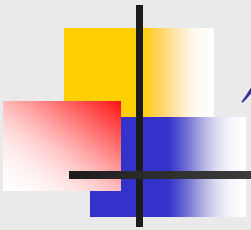
信息系统的基础概念

- 信息
- 数据
- 实体，属性和属性值
- 知识



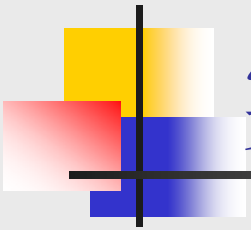
数据

- 数据是符号。
- 是人们用来描述客观事物、记录客观事物属性的可以鉴别的符号。
- 数据具有很多种形式。文字、数字、图形、声音、影像都是数据的形式。



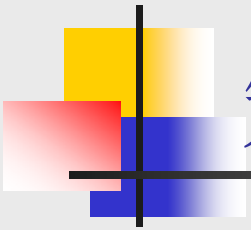
信息

- 信息是有意义的数据，是对决策有帮助的数据。
- 信息是经过加工的数据，是具有意义的数据，并且还要对决策有帮助。因此信息一般具有时效性和目的性。



实体，属性和属性值

- 实体，为最一般的模型，最没有特点。
- 属性，对实体的描述，对实体的特殊化。
- 属性值最终确定了实体究竟是什么。



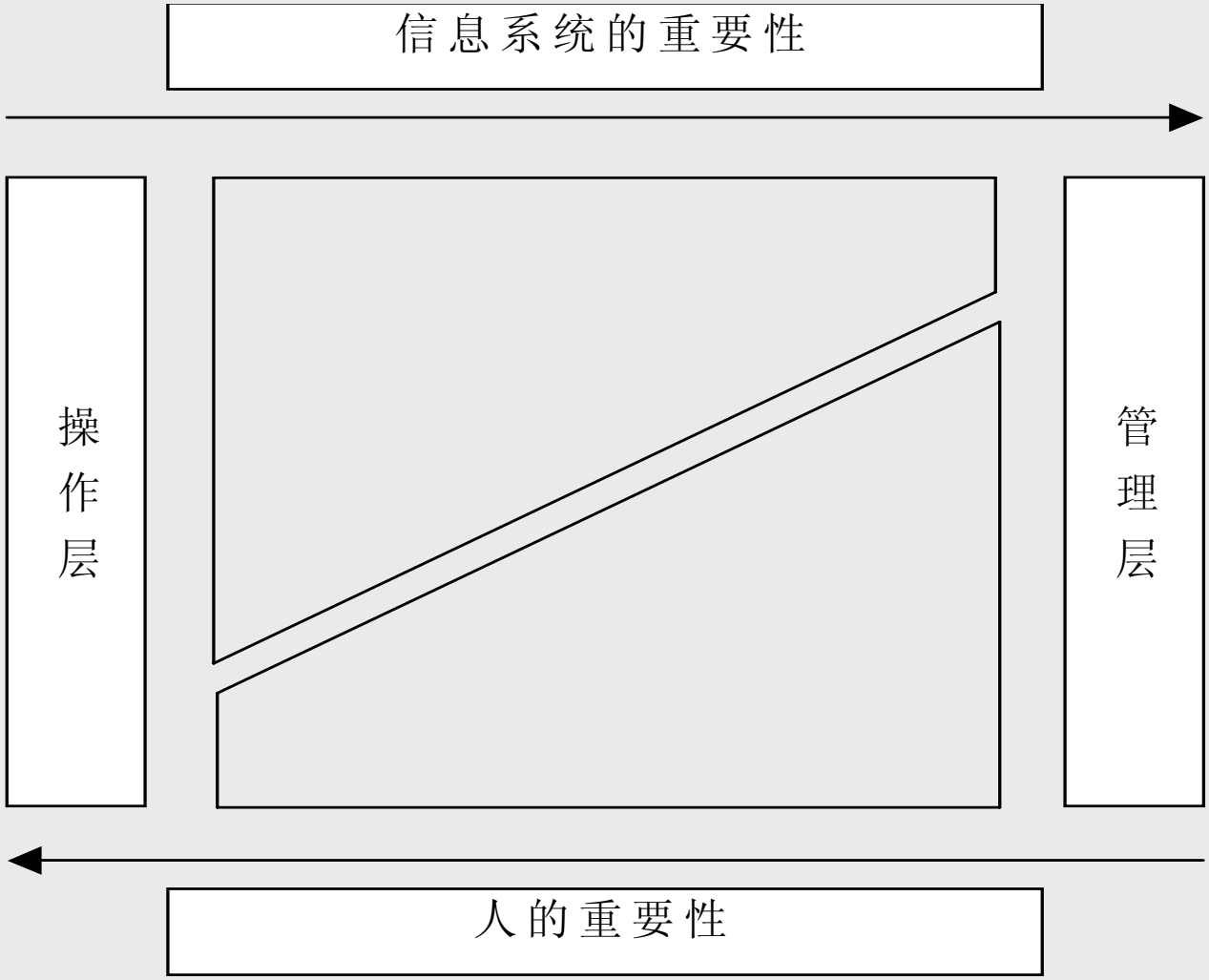
知识

- 知识是人们对实践进行总结而得到的结果。
- 知识包括：做什么的知识，如何做的知识，这些是容易编码处理的知识；怎样做的知识、谁来做的知识，这些是不容易编码处理的知识，一般来自于人们的归纳和总结。

信息系统的分类

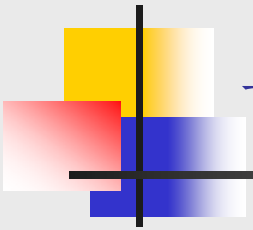
- 六种不同类型的信息系统。
- 交易处理系统（TSP）。负责企业最基层活动的相关信息的记录和处理。
- 办公自动化系统（OAS）。用来汇集操作层输出的信息，进行加工提炼，例如通过汇总、统计便于这些信息在组织内部共享，或者进行进一步处理。
- 知识工作系统（KWS）。用来帮助企业中的一部分员工生成新的信息和知识。例如CAD系统、软件开发系统。
- 管理信息系统（MIS）。对组织内部信息进行汇集、共享，以及进行范围有限的计划、控制和决策：一般考虑的范围都局限在组织内部，时间主要局限在过去和现在。
- 决策支持系统（DSS）。同时考虑组织内部和组织外部的信息，主要面向将来，对组织的决策提供直接支持。用来解决半结构化问题。
- 经理支持系统（ESS）。在高级管理层。为解决非结构化问题提供支持和辅助，而不是直接参与解决非结构化问题。

企业的组织和企业中人与信息系统的相对重要性关系



信息系统的开发是一项复杂的系统工程

- 企业信息系统建设往往不可能一步到位地完成，需要分阶段开发实现。
- 不管怎样，都必须在开发之前将整个系统当作一个有机整体进行数据规划和设计，否则会造成一系列的不良后果：数据冗余、处理错误、系统的高复杂程度和低维护性、管理的混乱。



计算机只能识别二进制数据

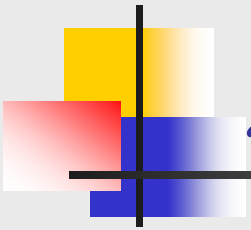
- 但是现实问题的数据绝大部分不是二进制数据。那么数据是如何输入计算机并且在计算机中处理后输出的呢？
- 数据输入：这是一个数据制式的转换过程。

计算机中的数据转换

- 数字是直接转换的。
- 文字用一个代码对照表转换成为数字。
- 图形利用排列组合原理+颜色对照表转换成为数字。
- 声音利用模数——数模转换成为数字。
- 影像利用数据压缩方法转换位数字。
- 最终在计算机中处理的都是二进制数据。

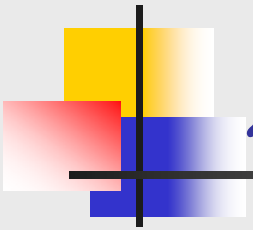
数据的处理

- 计算机依靠硬件和软件的结合来进行数据处理，而且一般都用非常“愚蠢”的方式处理。
- 原因：
  - 1、计算机硬件必须在程序的控制下才能够工作。因此从这个意义上讲当代计算机的所谓“智力”完全来自计算机的程序设计师
  - 2、计算机通过运行程序，也并不能够具备人们通常拥有的直觉、想象力之类的能力，因此它解决问题就只能够依赖它的强项：处理速度快，处理信息量高



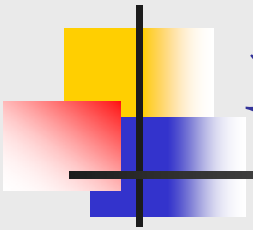
总结——计算机能做什么

- 计算机适合完成大量的、精确的、重复的数据处理和计算工作



总结——计算机不能做什么

- 计算机不适合完成需要直觉、想象力、归纳推理能力的任务，例如进行科学发现和制定企业管理的各种中、长期决策。



决策过程的中心在于决策者

- 决策支持系统对于决策只起到辅助的作用。决策支持系统是一种信息系统，它是用来辅助决策者进行决策的。

决策支持系统的定义

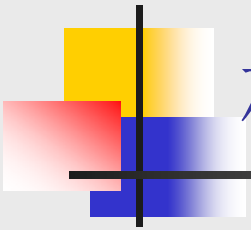
- 是一种信息管理系统；受控于一个或多个决策者；面向非结构化或者半结构化问题中的结构化部分，以改进和提高决策结果的质量为目标；作用是对决策者提供辅助。

决策支持系统具有的一般特性

- 用于半结构化或者非结构化问题的决策。
- 无法取代决策者，在决策过程中起辅助决策者的作用。
- 决策过程仍然由决策者完全控制。
  - 应用决策支持系统的目的是要提高决策的效果：让最终形成的决策更好。
- 以数据和相应领域的模型作为决策辅助的基础。

相关概念：问题的结构化程度

- 判断面对的问题的结构化程度的高低，可以根据三个标准判断：
  - 决策要实现的目标之间是否存在冲突，冲突的程度如何；
  - 可以选择的备选方案数量多少，方案与方案之间的界定是否清楚；
  - 决策带来的影响是否能够确定，如果不能够确定那么不确定程度有多高。
- 面对半结构化问题或者高度非结构化问题，决策支持系统对于决策提供的帮助在于：在问题的“结构化”部分为决策者提供有力支持，让他们能将更多的精力放在问题的非结构化方面，也就是人能够发挥长处的地方。



模型

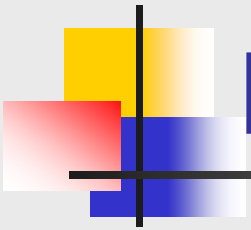
- 模型是对真实问题（真实现象）的简化；另外，模型只有在人们了解了足够多的真实现象规律之后才有可能建立，因此模型同时也是真实现象（真实事务）规律的反映。正因为如此，利用模型可以对真实现象的未来趋势进行预期。所以试图通过预测未来情况来为决策者提供辅助的DSS，它的功能的实现非常依赖于模型。

DSS的能力限制

- DSS能够做到：借助于计算机强大的计算和信息处理能力，高超的速度，和巨大的信息储存能力
- 扩展决策者处理信息和知识的能力
- 扩展决策者解决大规模、耗时而且复杂的问题的能力
- 缩短制订决策需要的时间
- 增强决策结果的可靠性
- 鼓励决策者进行探讨和钻研

DSS不能做到

- 对人能力的模仿，例如创造力、想象力、直觉等
- DSS本身的设计方案以及所拥有的数据和知识的限制，超出了这些限制范围之外的决策辅助工作是无法完成的。
- 用户（命令）界面的限制，例如无法提供利用自然语言方式的指令输入。
- 缺乏通用性。DSS一般都是面向具体专业方向的特化系统，因此在面对综合决策的时候无法提供很大的帮助。



DSS的分类

- 根据DSS支持偏向的不同，可以分为面向数据的DSS和面向模型的DSS。
- 根据DSS解决的问题的属性，可以分为一般性的DSS和专门性的DSS。
- 根据使用DSS的方式不同，可以分为程序化和非程序化的DSS。
- 根据对决策者人数的支持，可以分为单用户DSS与群体用户DSS。

决策支持系统的起源

- 根据20世纪70年代的一项研究，发现管理决策模型之所以不流行是因为管理者缺乏快速、可靠地应用这些模型的能力。借助于当时的微电子革命，就有人设想利用计算机的强大信息处理和计算能力来将这些模型实用化，这就是DSS的起源。

决策支持系统的组成

- 五大部分：
  - 数据管理系统
  - 模型管理系统
  - 知识引擎
  - 用户界面
  - 用户

数据管理子系统

- 在DSS系统中数据库管理子系统管理和某一特定决策相关的数据。提供对数据使用的支持和对数据安全、完整性的支持。数据库子系统还可以划分成多个子系统，例如：数据库、数据库管理系统、数据仓库、专门的检索查询工具。
- 在企业运营过程中积累的数据是企业的一种重要资产，这些数据是企业进行相关决策的基础。这些数据应该用有意义的逻辑格式加以保存并且同时保存数据之间的逻辑关系。
- DSS系统中数据的来源。有三种来源：来自于组织（企业）内部；来自于组织外部（有专门的数据传输服务商提供很多人都需要的外部数据，例如总体经济指标数据、信用数据等）；来自于用户（这个来源的数据被用来进行所谓的“用户个性化设置”）。

模型管理子系统

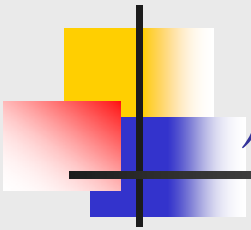
- 在DSS系统中模型管理子系统提供对需要使用的定量模型的保存和使用方面功能的支持。
- 在DSS中应用模型进行决策辅助是成本——收益权衡的结果。对于决策而言，只要能够获得精度足够的预期就可以了。
- 模型库在逻辑上非常类似于数据库，模型库管理系统提供的功能也非常类似于数据库管理系统提供的功能。模型库管理系统提供的功能有：按照一定的格式保存模型；根据用户的要求操作模型，包括整合和执行模型、建立新模型、对模型进行查询、修改、排序等。

知识引擎子系统

- DSS的知识引擎子系统提供对推理的支持。推理是决策必不可少的组成部分。包括两个部分：知识库和推理机。
  - 推理是利用现有信息推导新信息的过程。现有的信息也可以是以前推导出来的信息。
  - 知识库中记录进行某个专业方向决策推理所需要的知识。这些知识简单地说可以划分为两大类：事实和规则。注意：进行推理依赖于哪些事实也是重要的知识。
  - 知识库中记录的知识一般都与特定问题相关，也就是说这些知识的应用范围都很狭窄。
  - 推理机。这是DSS的用来根据提出的给定问题检索相关知识并且进行推理，获得需要推理结果的功能模块。

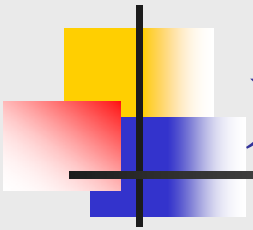
用户界面

- 通常DSS的用户界面被认为是系统中最为重要的部件。这是因为对于决策者（用户）而言，用户界面和整个DSS是等同的——拙劣的界面意味着拙劣的系统，反之亦然。
- DSS用户界面试图实现的目标有两个：
  - 用户访问简便性
  - 界面通用性。



用户

- 狭义的DSS用户的定义是：利用DSS的决策辅助功能进行决策的决策者；广义的DSS用户的定义是：与DSS之间有直接数据交流的人。无论哪种定义，“用户”在DSS的整个生命周期中都积极参与其间。
- 不同的决策者使用DSS的模式不同。可以划分为四类：订阅方式、终端方式、职员方式、中介方式。
- 广义上看有四类用户可能和DSS有直接的数据交流：决策者、中介、维护员（或者操作员）、输入员。

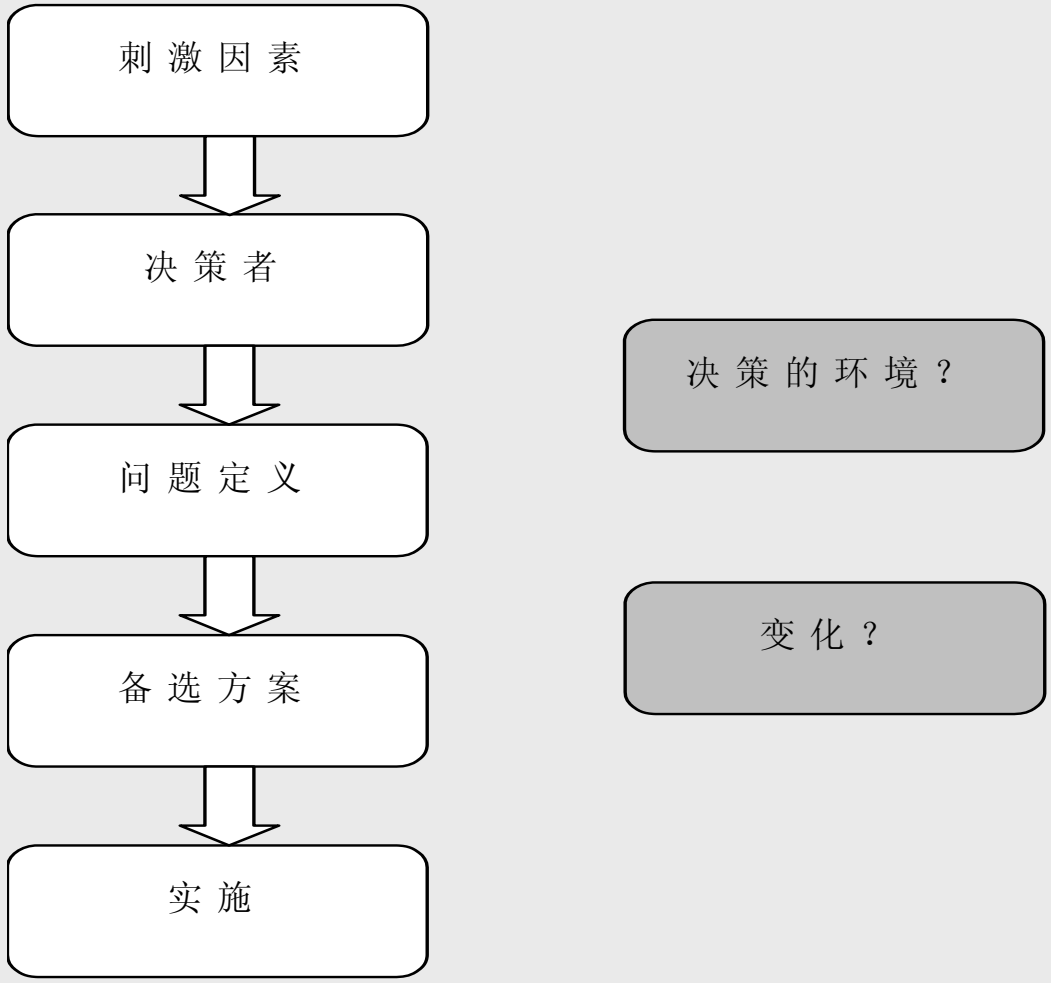


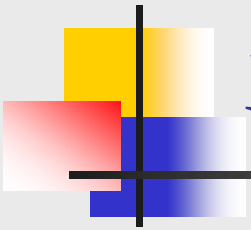
为什么要关心决策者

- 决策并不是简单的处理大量数据、运用很多定量模型、能够根据一些知识和推理规则进行有限的推理就能够解决的，最终这是决策者要完成的工作。因此DSS提供的辅助应该是决策者决策所需要的。
- 需要了解决策过程以及决策者的相关情况，才能设计和开发出能够真正为决策提供辅助的DSS。

决策是如何进行的——一个决策模型

- 决策由这样一些步骤组成





决策模型（续）

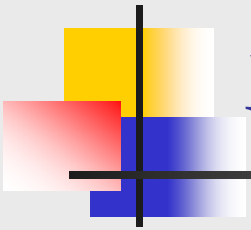
- 这个模型仍然与真正的决策有差距，它没有考虑两个重要的影响决策的因素。首先决策是动态的；其次决策一定要考虑环境因素的影响（或者限制）。因此这个模型只适合于用来进行说明。
- 通过考察制定决策的步骤，我们可以大致了解在什么地方需要DSS的辅助；以及我们希望DSS能够起到什么样的作用。

刺激因素

- 刺激因素是产生整个决策过程的原动力。刺激因素是一些外部因素或外部力量，它们让决策者感知到存在问题，进而打算做出决策以解决这些问题。

问题

- 在这里问题可以被简单地定义为对事情当前状态和期望状态之间差距的感知。
- 在很多情况下问题并不具备外在的表现形式。
- 认为刺激因素是外部因素，是因为就我们这里的目的而言，我们并不关心它们到底是什么，这样就可以把它们看成是外部因素来简化问题。

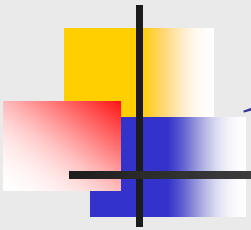


决策者

- 在这里决策者可以被视为是一个功能“黑箱”。因为一方面我们不了解决策者进行决策的具体机制；另一方面对此我们也没有兴趣。我们只关心“决策者”需要哪些输入以及会产生哪些输出。

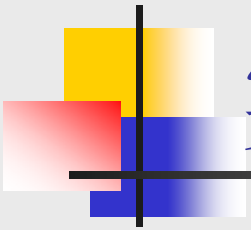
问题定义

- 想通过决策解决问题，必须首先有明确问题是什么，这就需要定义问题：界定决策的目标（要解决什么问题）并且把它明白无误地表达出来。
- 错误的问题定义或者不明确的问题表述都可能给后续决策和实施带来麻烦。
- 在这里DSS能够帮助决策者识别正确的问题定义，以及尽量清晰地描述问题。



备选方案

- 生成可以考虑的备选方案并且从中挑选最为有效的备选方案，这是决策的核心。
- 这里是DSS能够给决策者提供最多辅助的地方：根据模型提出可以考虑得备选方案、提供定量化方法分析各种备选方案的优劣、帮助决策者科学选择最优方案。同时这里也是以前决策者感到最为吃力的地方。



实施

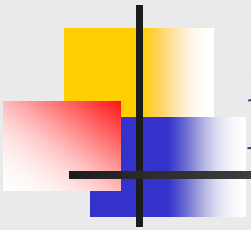
- 这实际上已经不是决策的有关步骤了，但是一个决策如果不实施那么就毫无意义。
- 这个阶段的工作绝大部分由人来承担，DSS能够提供的帮助非常有限。

根据决策者组织类型进行的分类

- 不同组织类型的决策者制定决策的方式不同，因此要求DSS提供辅助的方式和辅助内容都有所不同。

个体决策者

- 个体决策者单独行动，独自承担从分析信息确定问题、分析数据、应用模型并且最终决策的整个过程，当然也完全掌控整个决策过程和结果。
- 因为个体决策者对于决策过程的完全控制，因此个体决策者本身的特性：知识、能力、经验、个性以及偏好等都能够直接影响最终决策的制定，以及他在决策过程中需要什么样的辅助。

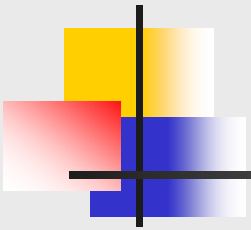


多人

- 在这种情况下决策由多个决策者相互作用来完成，每个决策者都能对最终决策的形成施加一定影响：通过达成一致意见或共同约定的方式做出。
- 决策者之间完全独立，决策者之间不存正式的组织关系：这些决策者每个都可以有自己的动机，从自己的角度进行决策；另外他们可能使用相同的DSS，也可能使用不同的DSS。
- 多人决策的时候决策过程中并不存在正式或者公开的讨论过程，制定决策需要的交流一般是在制度层级或者职权、等级的帮助下实现的。

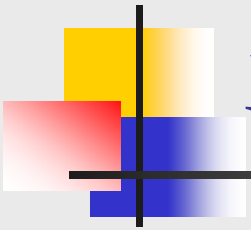
群体

- 群体成员之间存在一个正式组织结构，群体的每个成员对决策都有相同的发言权和影响力：没有人对于决策具有决定性的特权。群体决策的整个过程都是在正式的程序和步骤下展开的：通过例会、各种日程安排和议事程序进行。
- 群体决策者的例子如评审委员会。



团队

- 团队结构是个体与群体的混合。在团队中决策的决定性特权掌握在某个个人手中，但是他的决策受到团队中另外一些“助手”的支持和影响。
- 这些助手起的作用可能只是简单的收集和整理信息，但是也可能是提出参考方案或者分析参考方案这样的重要作用。
- 团队决策的实质是一种受影响的单方决策。

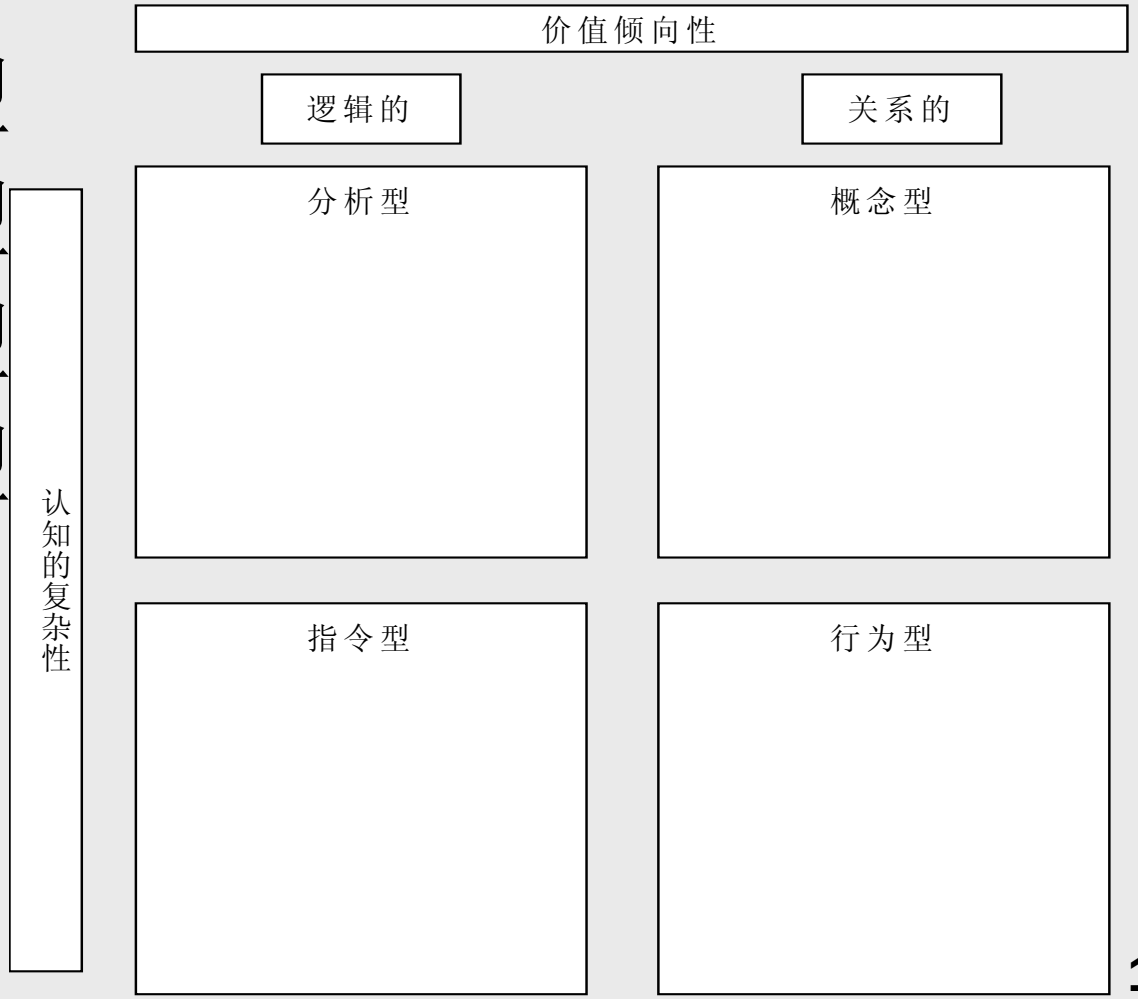


决策风格

- 决策风格是影响决策者决策行为的一个重要因素。简单地说决策风格是对于决策者决策方式的描述。
- 决策风格由这样一些因素决定：问题的背景、决策者的感知和个人在特定环境下的价值观。

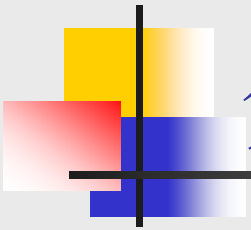
可以从“认知的复杂性”和“价值倾向性“两个维度对决策风格进行分类

- 指令型
- 分析型
- 概念型
- 行为型



指令型

- 指令型风格的决策者不能忍受或者感知含糊的问题背景；强调从逻辑结构出发进行决策；这样的决策者并不需要大量的数据，也不会考虑大量的备选方案；他们通常是追求效率和反应速度的决策者：要尽快取得结果；这种类型的决策者一般被认为口头交流能力比其他交流能力，例如写作能力要强。



分析型

- 分析型风格的决策者能够忍受和处理问题背景中的高度模糊性；强调从逻辑结构出发进行决策；这样的决策者偏好在决策中使用大量的数据，考虑尽可能多的备选方案，关注细节，因此决策需要比较多的时间；他们偏好而且擅长的是应对挑战，处理新的、常常是未预料到的背景下的问题；以解决问题为乐；相比较而言，他们的笔头交流能力比较好。

概念型

- 概念型风格的决策者能够感知和忍受问题背景中的高度模糊性；强调从人与人之间的关系角度出发分析问题进行决策；思维方式开放跳跃；从人本或者人文的角度进行判断；喜欢参与和放松控制；很多是“动口不动手”。

行为型

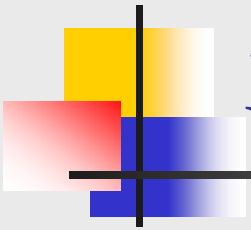
- 行为型风格的决策者不能够感知和忍受问题背景中的高度模糊性；从人与人之间关系的角度出发进行决策；基于有限的数据分析或者没有数据分析，因次很多时候现的“决策不科学”或者“目光短浅”；喜欢沟通，善于说服，是情绪化的和冲动的决策者。

对于决策风格分类的说明

- 人是非常复杂的，绝对不能认为决策者只具有一种决策风格。但是认为决策者具有一种主要的决策风格还是可行的，这样也便于我们设计和开发“用户导向”的DSS。

研究决策风格对于DSS设计和开发的帮助

- 让DSS提供的辅助功能能够和决策者互相配合，取长补短。



决策的有效性

- 如果在问题给定的背景下（环境限制下）决策达到了原先预定的目标，那么就可以说我们做出了一个好决策；另一种理解是：如果决策解决的要解决的问题同时并没有产生新问题，那么它就是一个好决策。
- 注意：决策的好坏评判一般只能够事后做出。对于决策质量的判断从来就没有什么固定的规则；也不存在总能够改进决策成功机会的金科玉律。

总结

- 能够对决策有显著影响的一些因素。在典型的决策过程中，这样一些因素对决策者和决策过程都有显著影响：个人的和情绪的因素；经济的和环境的因素；组织因素；决策本身的背景。

个人的和情绪的因素

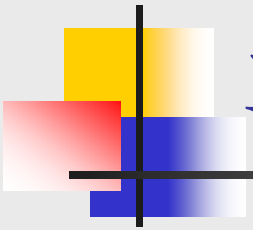
- 决策者也是人，因此他们作为人的属性：感情、健康、心理等方面的情况会影响到决策；作为人天然具有的认知方面的局限性也会影响决策。这些力量可能强化或者弱化决策者进行决策的能力。

经济的和环境的因素

- 这些因素包括资源限制、社会价值观和可能的政府管制。

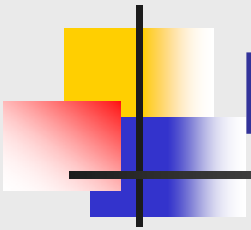
组织的因素

- 这些因素包括：组织对于决策的态度，组织对于决策所施加的限制，组织文化等。



决策问题本身的背景

- 这些因素中最为突出的可能就是决策时间方面的限定了。



DSS如何帮助决策

- 增强决策者处理复杂问题的能力。
- 从多个角度研究问题。
- 生成多个高质量的、供考虑的备选方案。

为什么决策如此困难

- 一般都认为进行决策是一项困难的工作。
- 导致决策困难的因素既有外部的也有内部的。

决策困难的原因可以归纳为四个方面的因素

- 决策要解决的问题结构方面的原因：程序化决策问题和非程序化决策问题。
- 认知的局限性。
- 不确定性。
- 决策的多重目标和多重备选方案。

决策要解决的问题结构方面的原因

- 程序化决策问题和非程序化决策问题。
- 结构化决策问题反复发生，具有现成的决策程序；非结构化决策问题很少出现或者以前从未遇见，不存在现成的决策程序。
- 结构化决策问题我们知道如何入手解决；而非结构化问题一开始我们根本无从下手，要经过一番研究和判断才能够理清头绪，找到入手点。
- 因此解决这类问题笔程序化决策问题要困难。一般意义上“决策问题”都是半结构化或者非结构化问题，并没有一个能够完全依赖的决策程序存在。

认知的局限性。

- 人脑能够同时接受和处理的信息数目是有限的，这就是所谓的认知方面的局限性。因此人们习惯于将信息分类、聚集成大块，以此来部分地弥补这个缺陷。

不确定性

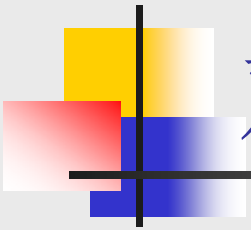
- 在确定的环境中决策当然容易，但是在多变的、不确定的环境中决策就困难多了。

决策的多重目标和多重备选方案

- 因为在目标之间、备选方案之间，以及备选方案和目标之间存在相互影响和相互依存的关系。所以在决策选择的时候，需要对每一个备选方案都进行详细分析。因此每增加一个目标、每增加一个备选方案都会增加决策的困难。

决策的分类

- 可以根据很多标准对于决策进行分类。
- 按照当前流行的决策分类标准，可以这样对于决策进行分类：
  - 基于协商的分类，分成：常规型、创造型、协商型。
  - 基于行为的分类，分成：创业型、适应型、计划型。
  - 基于策略的分类，分成：计算型、判断型、折衷型、灵感型。



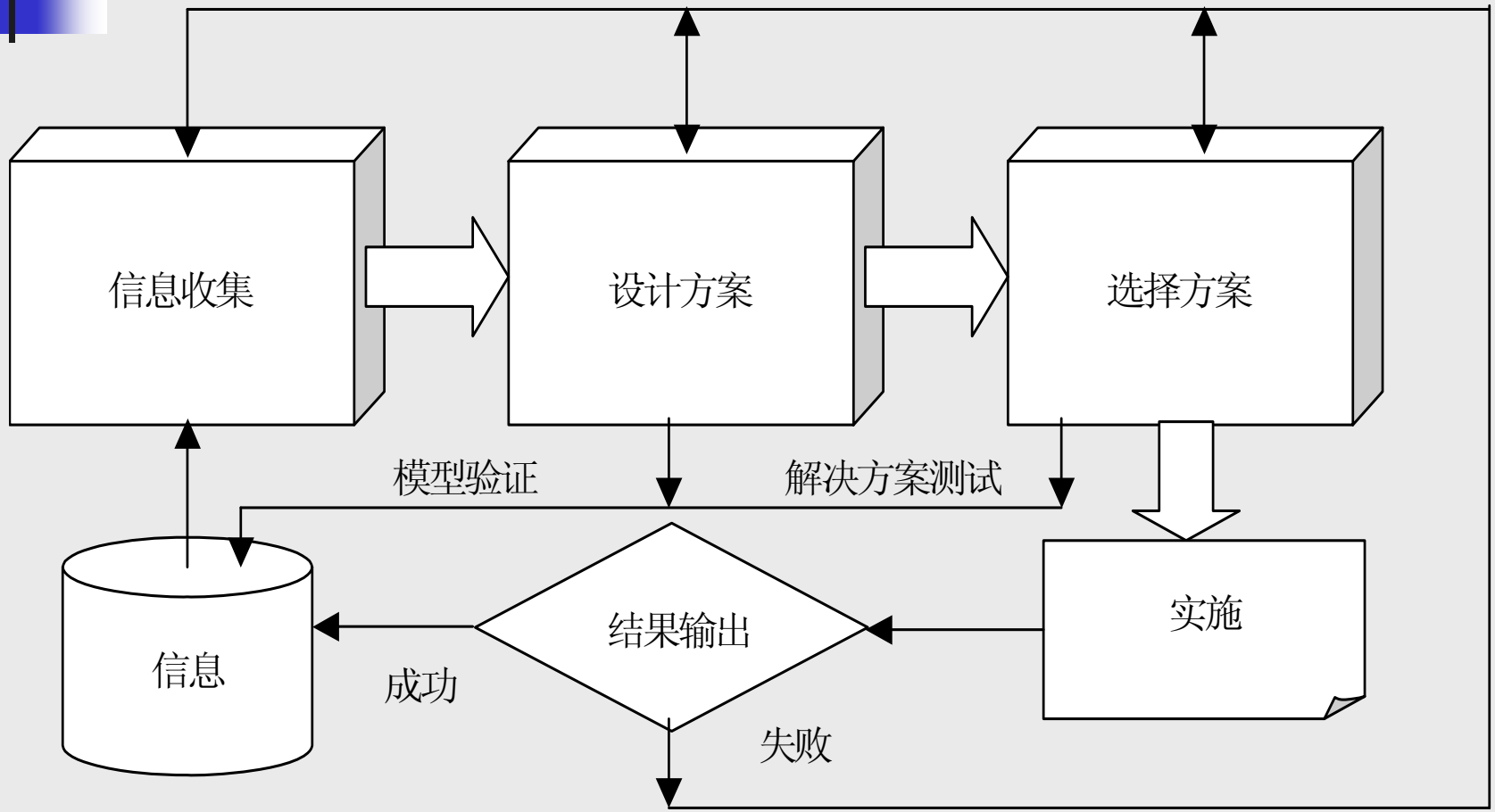
对决策进行分类的目的

- 通过更好地理解对决策进行分类的依据，也就能够更好地确定对特定类型的决策，具有什么样的属性的决策支持系统能够最好地提供支持。

决策者如何进行决策

- 这是关于决策者这个“黑箱”如何行使决策功能的理论。

Simon的问题解决的三阶段模型

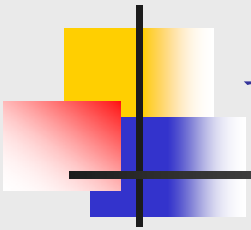


决策也是解决问题的一种具体实例

- 根据Simon的模型，解决问题的过程可以被描述为一个事件流，它既包括直线式解决问题流程也包括环形（反复）的解决问题流程。

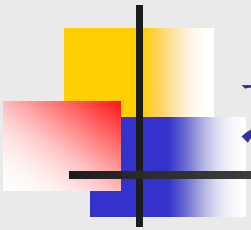
信息收集

- 这个阶段是决策的开始阶段。
- 信息收集阶段有三个任务：
  - 1、确定问题是什么；
  - 2、确定问题的“所有权”：问题是否在决策者可以解决的范围内。如果问题在决策者可解决的范围之外那么这个问题就不是一个决策能够解决的问题，而是一个“环境因素”或者“环境限制条件”。对这样的“问题”，不会有后续决策行为；
  - 3、为后续决策工作的展开收集信息。



设计方案

- 这个阶段要形成用来解决问题的各种可行备选方案，并且分析这些方案的可行性以及成本和收益。
- 这是决策活动的关键阶段。生成的被选方案数量的多少以及对这些方案进行分析的结果对决策结果有重大影响。
- 决策者很可能因为要做出这些子决策的缘故而要返回上一个阶段，收集更多的信息。



选择方案

- 从各个备选方案中，根据某种规则选择合意的那个作为决策结果。
- 这似乎很简单。但是环境中存在不确定性，立足于当前环境或者根据当前环境所做出的预期很可能和真正实施时的环境相符合。因此，决策总是有风险的，同结构化决策相比较，半结构化和非结构化决策的风险要高得多。

作为理性人的决策者遵循的原则以及选择决策的最终结果

- 传统上理论中的理性人的定义：完全理性，完全信息。
- 最优化策略是这样的理性人进行决策所遵循的唯一策略。最优化策略认为决策者的决策结果应该是从所有可行方案中最优的那一个。最优这个概念就是数学中的“全局最优”，让某个属性达到最值，例如价值最大，成本最小等。

传统上对最优化策略的批评

- 认为最优化策略缺少可实际操作性。原因在于最优化只针对定量属性才有意义，而管理决策中常常需要考虑的定性属性，因为无法量化因此对它们应用最优化策略的意义不大。

有限理性——理性人概念的颠覆

- Simon提出，因为成本方面的考虑，人们很少有兴趣获得“全部可行方案”；另外因为认知能力有限的缘故，就算能够获得全部可行方案，在决策时候他们也不可能完全理解这些方案，以及全面地比较这些方案。
- 因此，在理论中广泛使用的理性人概念在考虑实际问题的时候中并不成立。更合理的概念应该是具有有限理性的理性人。

在受限制理性下的决策策略：满意策略

- 在实际中决策者在集中精力寻找一种可以“接受”的方案——它能满足所有预设的需要，一旦找到这样的方案决策者就会采纳它，因此停止搜索完成决策。这样的结果当然不是最优解，至多只能够算是局部最优解。

在受限制的理性下，决策考虑的备选方案空间的变化

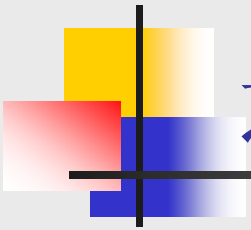
- 有限搜索策略。在现实中，决策者对备选方案的“全空间”并不感兴趣。他们需要的只是能够获得一个这个全空间的足够好的子空间，进而在这个子空间中搜索问题的解。因此决策的结果只能是一个相对而言最“满意”的结果。

受限制理性影响决策的另一个表现：问题和症状

- 虽然问题被定义为预期状况和实际状况之间的差距，但是并不是所有预期状况和实际状况之间的差距都是我们要解决的问题。其中很多只不过是问题所导致的症状。
- 解决了问题就可以消除症状，但是反过来并不成立。
- 因为有限理性的缘故，很多时候决策者满足于应付症状，反而错过了关注和解决真正的问题。

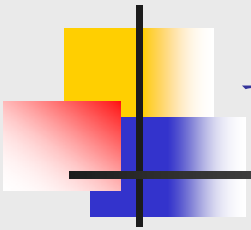
决策者的有限理性对DSS设计和实现的影响

- DSS也无法弥补决策者有限理性的缺陷，但是DSS可以帮助决策者扩大备选方案空间的大小以及提高有限搜索策略额效力。另外，DSS应该具备“进化”的能力——根据决策反馈逐渐提高生成和搜索问题空间的能力。



选择的过程

- 高度结构化的问题根本就不能够算是一个决策，因为一切都可以程序化，所以这样的问题并没有真正的选择。只有非结构化和半结构化的问题才要进行真正的选择和决策。
- DSS背后的世界观观点：我们可以从某种程度上预计未来。



认知过程

- 研究显示，人类决策者在认知的多个方面都存在局限。
- 分析认知过程，了解认知的局限能够让我们更加清楚决策者在决策过程中需要什么样的帮助，以及DSS能够提供哪些帮助。

导致认知局限的因素举例

- 短期记忆中只能够保留少量信息
- 决策者的智力水平
- 决策者闭塞的信念体系
- 决策者在处理信息能力方面的限制
- 决策者对风险的不同偏好
- 决策者的愿望。愿望不同对应的决策者希望的信息需求也不同

克服认知局限

- 决策者的确认识到这些局限因素，并且试图克服它们。对问题进行化简的做法可能是应对这些认知局限的一种重要方法。
- 在决策过程中服或者弥补这些认知局限的任务非常艰巨。对此DSS完全可以提供相应的帮助。

感知

- 感知是带有偏见的，决策者总是根据他们自己的理解和看法去感知，去选择看到什么，忽略什么。这种感知是对于现实的扭曲。
- 决策者的个人经验、目标、价值观、信仰等都对感知中的扭曲有影响。
- 感知的这种特性对于决策有深刻的影响。因为感知对现实的扭曲会影响到决策者决策的每一个阶段，从发现问题，到生成备选方案，乃至最终选择解决方案。

感知扭曲

- 感知扭曲造成的最常见的症状是导致决策者无法发现问题。
- 感知扭曲的表现形式之二：受限合理性。决策者会因为感知造成的偏见而被束缚住了手脚。表现在给一个问题强加上实际并不存在的约束和规定，因此人为缩小了考虑的备选方案的范围。
- 对此，DSS应该能够提供一定的帮助。DSS应该能识别问题的实际约束和人为给定约束，并对人为给定的约束提出质疑，帮助决策者克服思维的局限性。
- 感知扭曲的表现形式之三：只对已经习惯的东西感到满意，害怕冒险和变革。

判断以及认知局限对它的影响

- 在比较和评估各种不同的解决方案的策略中，判断策略很受欢迎。
- 在判断的过程中，决策者根据经验、感觉和知觉来进行选择，因此和详细全面地分析选择策略相比较，判断策略更加快捷方便，而且对决策者的压力也小。

但是判断很容易受到决策者认知局限的影响

- 判断主要依靠决策者对于过去发生过的事件的认知或者感觉。认知的局限性常常让这种感觉不可靠：
- 判断更加依赖于感觉和偏好，而不是分析和数据
- 直觉常常将决策者引入歧途
- 通过表象进行的判断往往不可靠
- 因此如果单独使用，那么判断往往只不过是猜测。一般判断策略要和其他策略综合起来应用才有效果。

决策中的“拇指规则”和启发式搜索

- 启发式推断规则（拇指规则）。这是一些在反复尝试的基础上建立起来的经验规则，一旦通过以前的经验证明可靠的话就会被认为是一种有效的推断工具因而反复地使用。
- 同穷举式搜索规则相比，根据启发式推断规则进行的搜索可以大大缩短解决问题的时候需要的搜索时间，因此启发式搜索往往更有效率。
- 例如，解决推销员问题的拇指规则就是：每前进到一个地点，就选择最近的下一个能到达地点作为下一个目的地。

应用启发式搜索规则中的问题：偏见

- 启发式推断规则往往失败在它们进行搜索过程中的表现出来的武断上，这表现在选择出发点、后续决策的顺序、论证最终结果是否最优等方面上。这种武断来自于启发式推断规则本身的经验性上。
- 具体表现：根据启发式规则，决策者往往高估低概率事件，低估高概率事件；倾向于对自己的预测能力过于自信；倾向于两两比较而不是从全体的角度出发考虑等。

有效性偏见

- 因为典型的决策者不能准确地评估特定事件发生的概率而产生的偏见。
- 这种偏见实际上是由于混淆了统计上的期望和随机事件的实际结果而导致的。

相关性错觉偏见

- 对于事件之间相关性的错觉：决策者因为两件事经常一起发生而错误地认为它们彼此高度相关。这种偏见会导致对决策结果的搜索偏离正确方向。
- 例如：体育彩票某个数字曾经出现在过去的中奖号码中和它将来会出现在未来的中奖号码中；股票投资家的成功与否和它股票投资能力之间的关系
- 对历史数据的追根究底式的回顾和分析可以有效的减少有效性偏见和相关错觉偏见；在缺乏这些数据的情况下，决策者至少可以试着诚实、客观地分析和评估一下自己的启发式规则，并且根据分析评估结果对这些规则进行一定的修正。

对于启发式搜索初始点的调整

- 设定初始点是进行启发式搜索所必需的，但是人们往往忽略了在后来的过程中对于初始点进行调整的需要，表现在他们不愿意调整他们的初始估计。
- DSS可以通过强迫决策者从区间的角度考虑问题，抑制用户将决策的结果固定在中间值上的趋势以提高决策的正确性。

代表性偏见

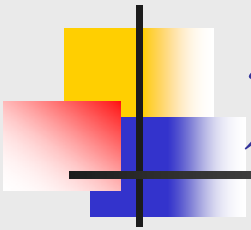
- 决策者往往根据小样本例子中得出的规律或者概率作为解决普遍问题的原则或者一般现象中的概率估计，这种决策往往忽略真实的规律或者概率预期。
- 例如，非常好的表现之后一般跟着的都是不那么好的表现；同时非常坏的表现后面一般跟着的也都是不那么坏的表现。

动机偏见

- 决策者的动机或者刺激因素，常常导致决策者只看他愿意看的，只想他愿意想的，这又引起了动机偏见。
- 这种偏见DSS难以帮助解决。一般可行的办法是给予决策者足够多的偏好不同的决策者的相关估计，让决策者能够自己警觉到这种偏见。

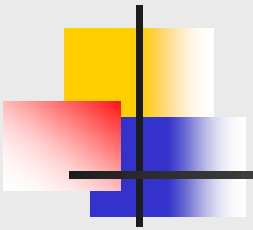
决策效果和效率

- 效果关注应该做什么，而效率则关注应该如何作。要想决策有效果，就需要不断的式英、学习和思考，付出进度缓慢成本高昂的代价；而要想决策有效率则只要简单地考虑如何最为经济地完成一件事情。
- 在决策效果和效率之间，通常是一种权衡关系。
- 例如，从效率的角度看，企业的任何研发行为都是无效率的：当前的经营行为并不需要研发，同时剔除了研法之后的确又能显著地增加当期利润。
- 经验表明，环境越是不稳定就越需要重视决策效果；相反的如果环境比较稳定那么就可以把相当的精力放在决策效率上。



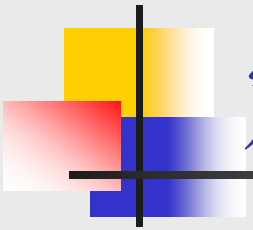
定义问题

- 全面而清楚地定义问题对正确的决策而言必不可少。这听起来简单，但是决策者常犯的典型问题之一就是忽略了给予问题一个全面的识别和定义，由此常常造成决策失败。



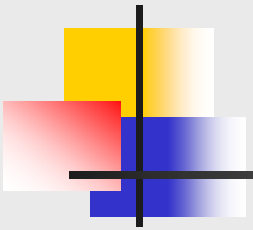
问题描述

- 根据问题的定义，一个清楚而且全面的问题阐述应该包括三个关键部分：
  - 对当前状态的描述；
  - 对期望状态的描述；
  - 能够作为区别当前状态和期望状态的标准的目标的描述。这个目标可以是单一目标也可以是复合目标。



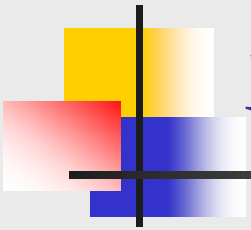
定义问题方面的错误

- 在问题定义方面决策者常犯的错误就是：过早地将注意力转移到解决方案的选择上，反而忽略了问题本身。
- 这种情况常常导致决策者得不到成型的问题描述，不明白要解决的究竟是什么问题，并且选择决策方案的眼光被限定在嵌入到问题描述中的那几个非常有限的方案中去。



问题的范围

- 定义问题之后决策者就必须检查和考虑，要解决问题需要遵循什么样的资源或者时间方面的限制和约束，另外还要考虑决策者的主观评价，例如优先权顺序。

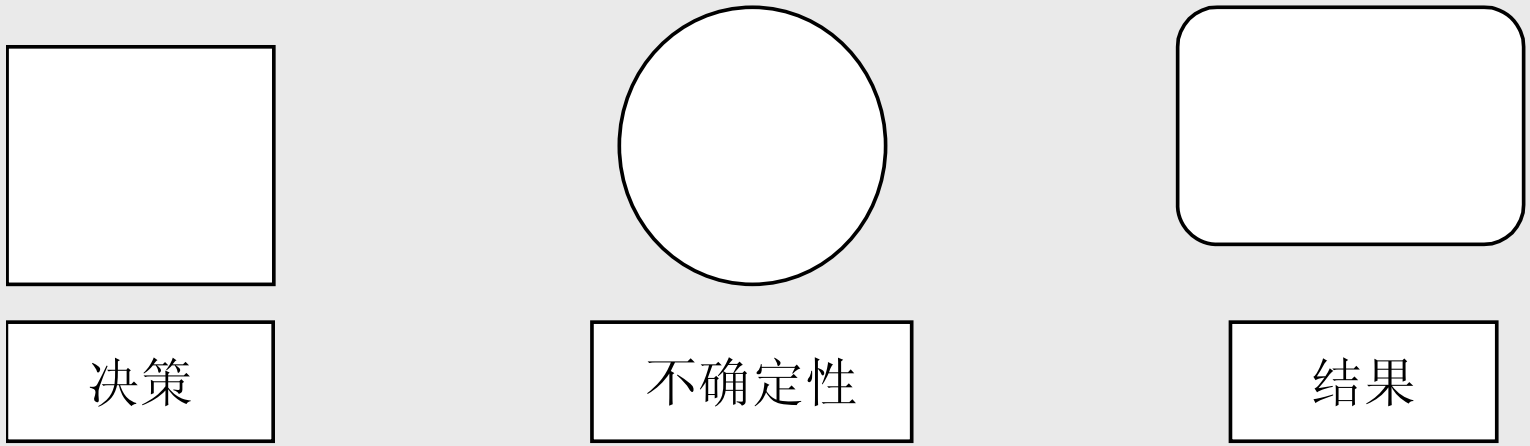


决策的结构

- 构建DSS需要对决策建模，这就是所谓的决策模型。
- 决策模型的三个组成部分：
  - 选择（决策）；
  - 不确定性；
  - 结果。

用“影响图”来建模决策

- 这种方法适于描述决策各个部分因果的结构；也可以用“决策树”来建模决策，这种方法适于描述决策选择和与概率有关的随机结果。

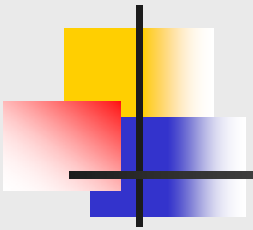


选择（决策）

- 决策可以选择不同的方案，因此导致不同的后果。决策必须有可选性否则就不是决策了，因此一般决策至少要包括两个备选方案，所以很多情况下“什么都不作”也被认为是备选方案之一。

不确定性

- 决策产生的具体结果总是受到不确定性的影响。对决策而言，选择和不确定性考虑的是两个阶段的概念。选择是决策必须要做出的行为，决策者必须从备选方案种选择一个或者几个作为决策结果；而选择产生的结果的不确定性则是选择之后才产生作用的。
- 不确定性是决策者无法控制的，决策者只能估计每种可能情况的概率，而选择是决策者能够控制的。



目标

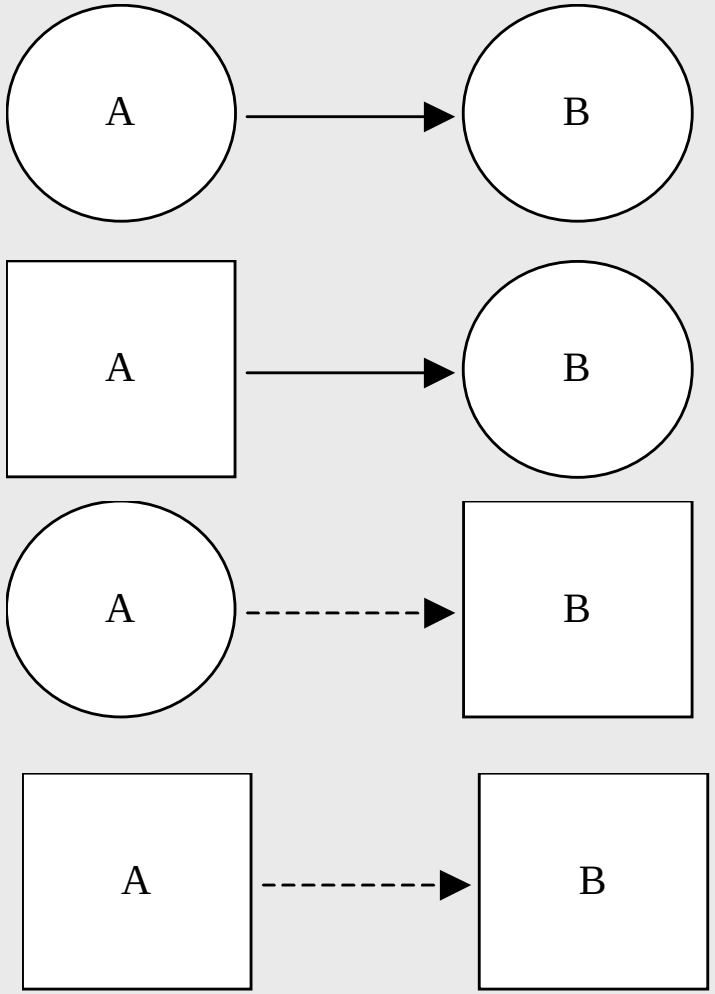
- 目标不同于影响图或者决策树中的结果。目标帮助我们分析和评价决策产生的结果是否为我们所希望。为了便于处理，目标应该是一种清楚并且能够将决策结果量化的评价标准。

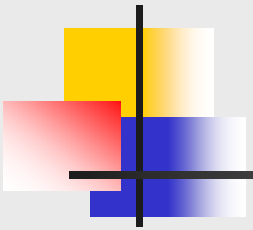
影响图

- 影响图由“决策”、“不确定性”和“结果”三种成份组成，用表示相互之间关系的实线箭头或者虚线箭头连接起来。
- 实线箭头总是指向随机事件或者结果，表明前后两者存在相关关系；虚线只指向决策，它表明决策（后者）是基于前者的基础上作出的。

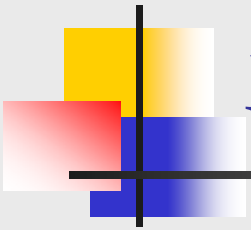
影响图的例子

- 表示随机事件B的发生受到随机事件A发生的影响。
- 表示随机事件B的发生受到决策A的影响。
- 表示决策B是在考虑随机事件A包含的信息基础上做出的。
- 表示决策B是在决策A之后作出的。



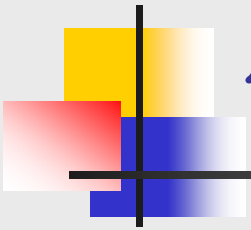


- 构造正确的影响图是没有回路的。不管从哪里出发都无法再回到起点。在影响图的世界里，没有后悔药。



决策树

- 决策树能够描述决策的相关细节，因此和影响图反映的信息相互补充。决策树能够描述的细节主要是关于随机事件、随机事件的概率以及随机事件的结果。
- 在决策树中没有箭头，节点之间只有简单的连接线，表示因果关系或者在时间上的前后相关关系。
- 从决策树的“选择”节点发出的分支表明各种可能的选择，要保证这些选择是单选的，因此如果决策需要做出多个选择，那么应该把它描绘成一个多阶段决策，每次都只作出一个选择。
- 从决策树的“不确定性”节点发出的分支表明随机事件的各种可能结果，要保证这些结果是互斥并且完备的（是随机事件所有结果的一个划分）。
- 决策树也是单向的。只能够从决策向各个结果前进，不能回到开始。



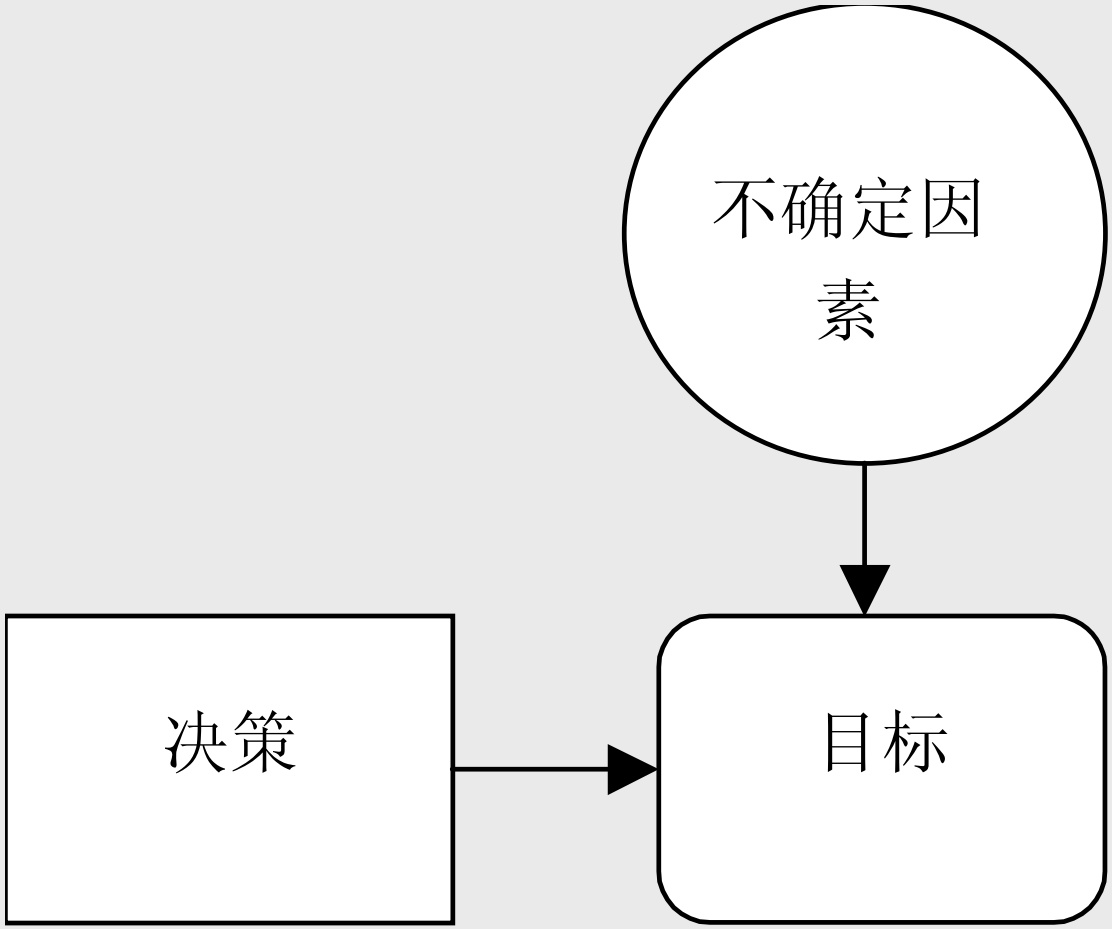
常见的决策结构

- 虽然决策问题不会唯一，但是了解常用决策的结构仍然是有意义的，这样可以帮助决策者快速地归类和定位当前决策的结构，为进一步分析提供一个正确起点。

基本风险决策

- 这时决策目标能不能实现要受到风险因素的影响；并且决策目标的实现只受决策选择和不确定性两个因素的影响。
- 例如，考虑预定在今年夏天开始的一种新口味冷饮的促销决策。决策结构由三部分组成的：选择：是否要促销以及不确定性：今年夏天的气候状况；目标：让新的冷饮取得成功，量化指标可能是在促销开始一定时间后要达到的日销售量；或者是要占领的市场份额。

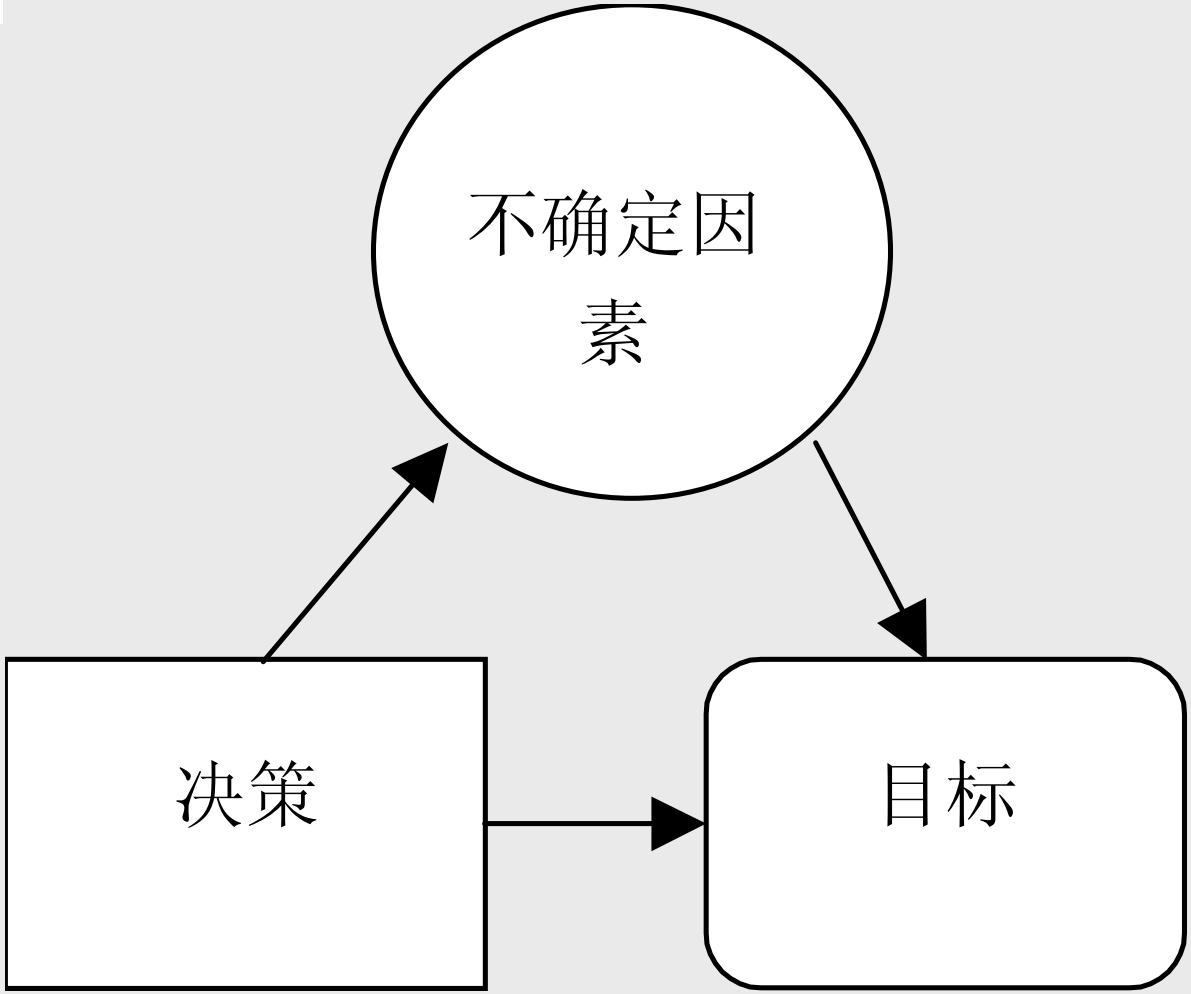
影响图



基本风险政策

- 这种情况是基本风险决策的复杂化。在种情况下决策选择不但对决策目标是否实现有直接影响，同时通过影响不确定因素间接的影响决策目标。
- 例如，考虑改换一种成功产品包装的决策。决策结构由三部分组成的：选择：是否要改换包装；不确定性：消费者对此的反应；目标：让这种产品更加成功。目标的量化指标可能是一定时间后要达到的日销售量；或者是要占领的市场份额。

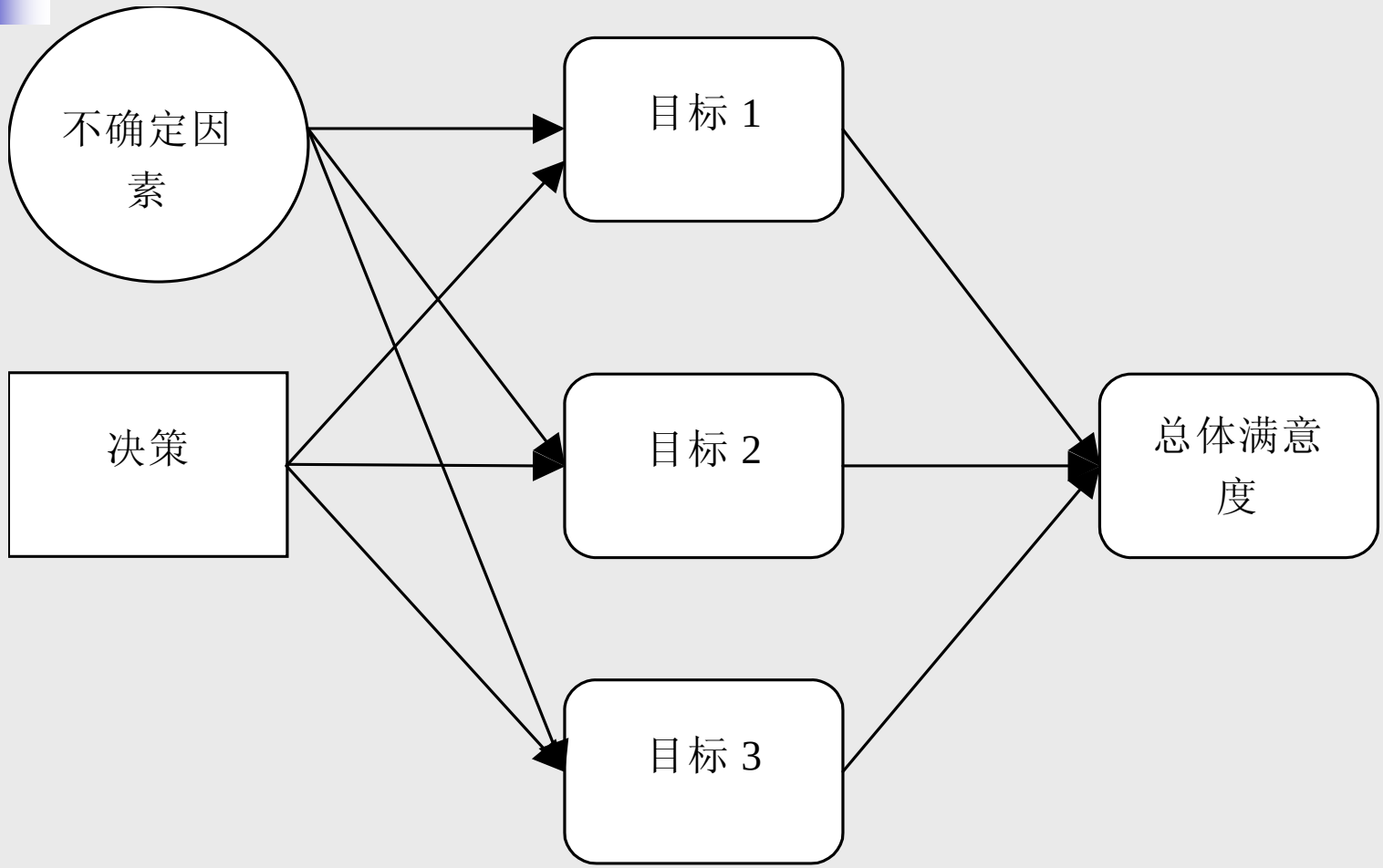
基本风险政策的影响图

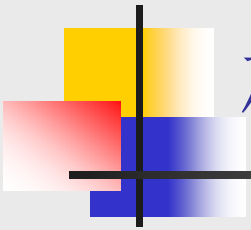


多目标风险决策

- 在这种情况下，决策要实现的是存在权衡关系的多个目标，因此决策是否成功要看综合这多个目标实现的一个总体满意度。
- 例如，考虑为了专门促销一种新型家用电器而组织产品管理队伍的决策。这也就是对这个产品管理队伍如何组织的决策，这种决策直接和多个目标相关，例如产品促销行动的开展是否顺利、产品促销的结果如何——比如产品的销售量和市场覆盖率、促销行动的成本等，最终决策是否合意要看决策能够实现的、在多个目标之间权衡一个总体满意程度；同时，风险因素也分别影响到了每一个决策目标实现与否，因此这是一个“多目标”决策。
- 因为多个目标之间关系复杂，所以解决这样的问题一般需要将决策分解成多个相互独立的子决策来分阶段完成。
- 如果决策选择不但影响到决策要实现的多个目标，还影响决策的风险因素，进而间接地影响到决策目标的实现，那么就是更复杂的一种决策结构，所谓的“多目标风险政策”。

多目标风险决策的影响图





确定性的多目标决策

- 在这种情况下，影响决策的不确定性或者不存在或者影响可以忽略，而如何在决策要实现的多个目标之间权衡，让决策的总体满意度最高则是重要问题。
- 例如，一个政府部门如何分配预算的决策问题。这种情况并不需要考虑什么风险因素，但是“最好的为民众服务”这个决策总体目标的实现却是要通过在许多具体目标之间进行权衡来做到，这些具体目标之间非常有可能是相互冲突的。
- 例如解决为了解决失业问题而进行的开发项目，就很有可能和政府的环保目标相冲突。

序列决策

- 很多决策是渐进式的分阶段决策：决策的环境和目标会随时间和决策进程的变化而变化，这样决策就变成了一个要让总体满意度最高的序列决策问题。
- 在这种情况下，所有的分阶段决策都是为了能够提高最终的“总体满意程度”。另一方面因为是分阶段决策，因此对于每一个阶段的决策，只要能够为足够多地提高总体满意成都就可以了。

决策模型的类型

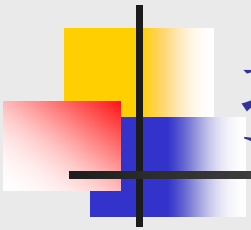
- 使用不同类型的决策模型作为具体决策问题的近似，不同类型的决策模型和不同的解决问题方法相对应。
- 可以根据考虑的时间因素来分类，这样就将决策模型划分成两类：静态决策模型和动态决策模型；
- 也可以根据构建模型依据数学还是依据逻辑来划分，这样就分成了抽象决策模型和概念决策模型。

抽象决策模型

- 这类模型致力于使用数学方法建立决策模型——利用数学工具预测决策的各种可能结果，使用数学方法解决决策问题。
- 这类模型一般都从数学上的最优化角度出发解决解决决策问题。

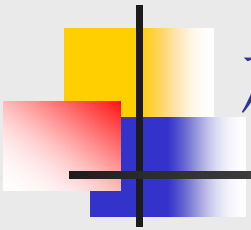
数学模型的近似合理性

- 利用数学进行建模就是在真实性和简化之间进行合理的权衡，因此和真实情况相比，模型总会丢失一些细节信息，模型给出的结论永远只是实际情况的近似。



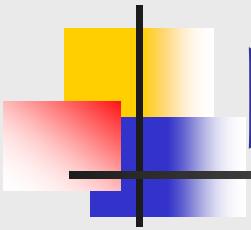
抽象决策模型分类

- 确定模型
- 随机模型
- 仿真模型
- 只适用于特殊领域的特殊模型



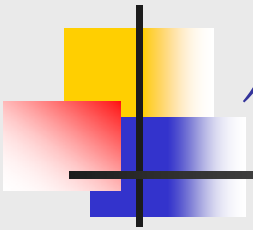
确定模型

- 对于这类模型而言，不考虑随机因素：相同的输入总是产生相同的输出。标准的确定模型有：最优化模型、线性规划、生产规划、财务计划等。



随机模型

- 随机模型中包括了不确定性的考虑。因为考虑了不确定性，因此随机模型的结构和处理都比确定模型要复杂。标准的随机模型有：博奕模型、排队模型、线性回归、时间序列分析等。



仿真模型

- 相当一些决策结构是复合的：同时包括了确定部分和随机部分，对于这种复杂决策可以用模拟仿真模型来处理。
- 模拟仿真模型可以看作是实验。这种实验用来测试一个复合系统在处于一个动态环境中的时候对环境变化的反应。

概念决策模型

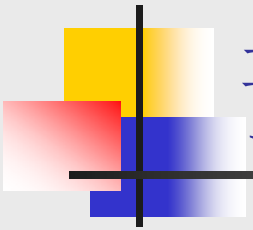
- 抽象决策模型并不总是适用的。如果决策者非常熟悉决策问题的结构，或者构造能够满足需要的抽象决策模型成本太高的话，用概念决策模型反而能够比较有利。
- 概念模型的建立非常依赖决策者的经验、直觉和判断。概念模型背后的世界观是：虽然所有问题都是独特的，但是没有什么问题是全新的。
- 过度主观的概念决策模型。从表面上看，概念模型的主观色彩也许的确比抽象模型要强烈，但是实际上很可能并不如此。

决策模型的清晰度测试

- 决策建模要直到模型结构的所有组建部分都已经被清晰地定义了为止。这可以通过对模型进行清晰度测试来实现。
- 模型的清晰度测试，具体地说就是对模型的每一个节点，确保已经尽可能确定了所有的对应的因素。例如对于“决策”节点，要确保备选方案的完备性；对于“随机因素”节点，要保证各个可能结果的完备性和互斥性，以及确定各个结果相应的概率水平。

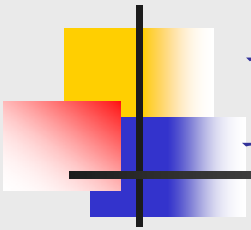
决策模型中随机结果概率的确定

- 如果决策过程中不存在不确定性，那么只要凭借逻辑分析和经验常识就可以解决所有问题，而决策问题也就成为机械问题了。决策模型中各随机结果概率数值的确定是利用模型解决决策问题必须的一个前提条件。
- 要能够处理不确定性，首先需要量化不确定性。在决策中不确定性的量化是通过应用概率概念解决的。



预测概率的技术

- 有很多用来估计概率的技术，在实务中常用的有线性回归、时间序列分析等等。这里介绍一些可以利用概率本身的特性来对概率值进行简单估计的技术。



直接概率预测

- 也就是猜测。当然如果拥有足够多的过去经验的话那么这种猜测可以相当接近于真正的概率值。

用构造无差异赌局的方式估计随机事件的概率

- 对于任意随机事件的概率估计问题，可以通过构造相应的一对互为镜像的赌局，让决策者或者概率估计者不断的调整他们愿意投入的赌金数额，使得他们刚好达到对于愿意/不愿意参加赌博持无所谓的态度（两者具有相同的期望）的方法来估计，此时通过对赌博金额的计算就可以很好的估计出决策者对于随机事件的主观概率。

用类比的方法估计随机事件的概率

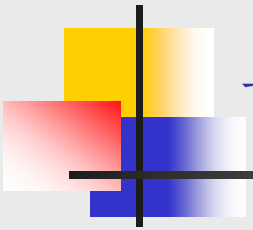
- 这种方式需要有一个概率已知的赌博作为参照系进行对比，用来估计另外一个赌博的选项所对应的概率的大小。

决策模型中的随机因素的简化

- 通过灵敏性分析可以帮助决策者简化相应的随机影响因素。那些单位变动导致程度轻微的重要影响；以及单位变动导致相当程度的不重要影响的随机因素可以简化为决策模型中的常量或者干脆省略。
- 灵敏性分析是情景分析的一种，通过给定一个情景（保持其它无关变量不变），改变要考虑的随机因素，检查我们所关心的被影响因素对于这个改变的反应，根据反应幅度的大小确定灵敏度。

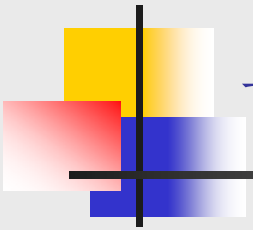
如何判断是否应该继续改进对不确定性的认识

- 精确的估计概率总是能够提高决策的效果的，但是这样做需要付出代价。因此是否应该进一步提高概率估计的精确度是一个成本——收益权衡的问题。



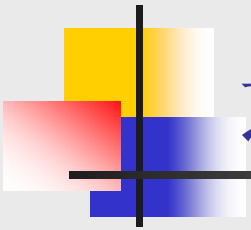
专家的定义

- 曾经有人给出过这样的专家定义：专家就是能够解决在狭小的专门领域中出现的所有问题的人。就“专门问题”而言这个定义有一些夸大，但是它正确的指出了专家是一个只面向特定领域的概念。
- 专家善于解决专门领域中的问题。一个专家必须是具备特定领域中相当多经验的专业人士。一般对于专家，只要给予有关领域问题足够多的信息，通常他都能提出一个相当好的解决方案。



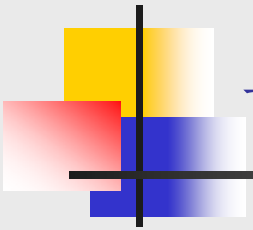
专家可能是一个年轻人吗

- 专家是如何具有这样不同于普通人的能力的？专家的能力来自于拥有的大量知识和根据这些知识进行的逻辑推理。逻辑推理能够帮助人们根据过去的事实和经验生成新的、很可能并不为他们所知的论断。



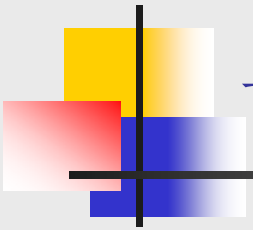
逻辑推理

- 推理：利用知识之间存在的逻辑规则以及对问题背景的了解产生原先未知或者事先没有明确表述的信息。
- 逻辑推理之所以有如此的能力，是因为在知识和知识之间逻辑关系。这些知识之间的逻辑关系也是知识体系中非常重要的一部分。因此，如果问题的条件和前提能够和某个逻辑关系的前提相一致，那么我们就可以应用推理方法得到问题的结论。



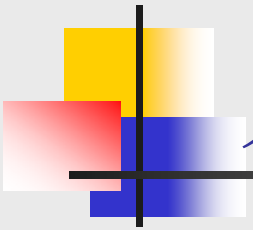
一个例子

- 例如：已知信息：John是Sam的儿子；Mary是Sam的女儿；John和Mary的母亲叫Anna，John是男性。
- 问题：Sam的妻子叫什么名字？John是个男孩还是女孩？Sam结婚多长时间了？
- 这里的推理是一个信息匹配的过程：用对问题事实的描述和作为规则的“如果——那么”进行匹配，一旦发现对问题事实的描述能匹配规则的“如果”部分，那么就可以顺理成章地说规则的“那么”部分也是成立的。这样我们就能获得有关问题的没有明确指出的事实或者新事实。
- 正是因为这是一种机械的匹配过程，因此开发叫做专家系统的计算机软件才有可能。



专家系统

- 专家系统是一种特殊的计算机软件。信息匹配是专家系统解决问题的时候所采用的一种主要方法。这种软件贮存了大量的特定领域的知识，利用适用于这些领域的专门规则进行推理。所有这些知识（包括规则）都是人类专家在研究和解决特定领域问题的过程中总结出来、通过一系列系统的方法采集和输入到专家系统中的。
- 专家系统可以作为真正的人类专家的助手，或者为一个不是专家的决策者提供专业的咨询，大大提高他解决特定领域复杂问题的能力。

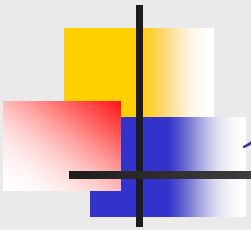


人工智能

- 专家系统应用了很多人工智能领域的技术。人工智能的目标是要让计算机能够像人类一样的思考、推理和学习，因此人工智能学科是一门计算机和认知心理学的交叉学科。

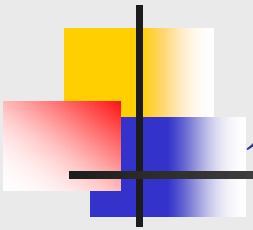
专家系统和人工智能的发展简史

- 专家系统和人工智能几乎是一同产生的。它们的根源可以追溯到20世纪50年代，美国智库蓝德公司卡奈基小组的研究工作。



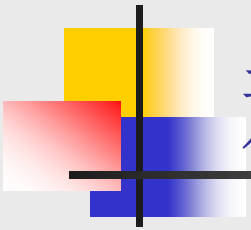
人们如何推理

- 人工智能要实现的目标是模仿人类思维和推理的方式，利用计算机重现这些思维和推理方式。因此人工智能的要点首先就在于对人思维和推理的研究和了解。
- 在心理学里面专门有一个分支：认知心理学来研究这个问题，对此也已经形成了一个完整而复杂的理论体系。在此我们只简单的介绍人们通常使用的推理方法，这些方法在专家系统和人工智能领域中也有直接的和大量的应用。



归类法

- 这是人们使用的最普遍、最直接的推理方法。
- 首先将各种事物根据它们自身的特征划分成不同的类别，然后再在相应的类别中按照具体属性的不同划分子类别，形成一种“树”型结构。
- 这些类别还可以划分层次。每一个层次的“类别”节点中都包含一部分关于事物属性的知识，这些知识是可以“继承”的——低层次的节点可以获得高层次节点的全部属性。
- 在归类体系的帮助下，只要定位了相应事物的类别和层次，那么根据分类和从属，很容易的就可以得知这个事物很多没有明确说明或者我们不清楚、不确定的属性。

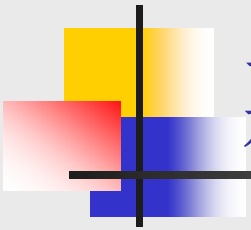


规则推理

- 利用给定的具有规范的“IF——THEN”格式的规则进行推理，这种推理的实质是匹配。规则推理应用的规则可以是明确给出的特定规则——例如税法和一般的法律规则；也可以是在启发式推理中采用的经验法则。这些经验法则可能缺乏正规性和严密性，但是在很多情况下可能是成本——收益权衡下的最优选择。

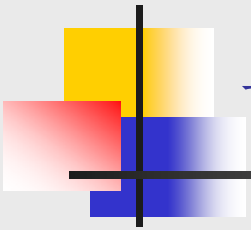
人们通常采用的推理方法是归类法和启发式方法的结合体

- 考虑问题的一般过程是：首先根据问题的特征对问题进行分类，然后将问题和人们过去经历过的事情进行比较，如果发现当前问题和过去某个经历过问题相匹配的话，就可以根据过去的问题的属性推导出当前问题的属性，根据解决过去问题的处理方法得到能够用来解决当前问题的备选方案。



期望法

- 一旦我们经历一种问题或者现象足够多的话，我们就开始期望问题已一定的方式出现，这样我们就可以判断是否发生了问题。根据期望进行推理实际上是一种简单的模式识别。

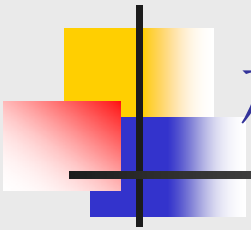


计算机如何推理

- 基于规则的推理
- 框架结构
- 基于案例的推理

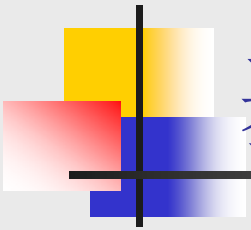
基于规则的推理

- 这是人工智能和专家系统使用的最为普遍的一种推理方法。
- 首先，向计算机输入对问题特征的描述，根据这些特征描述计算机搜索知识库中的规则，看看哪些规则能够和这些描述相匹配；如果找到能够匹配的适用规则就根据这个规则得到相应的结论。
  - 规则包括两个部分：前提和作为前提结果的结论或者改变状态的操作。这种规则一般以IF——THEN的形式出现：IF前提条件——THEN操作或者结论。
  - 搜索就是寻找看有哪些IF——THEN规则的前提条件能够和对当前问题的描述相匹配。



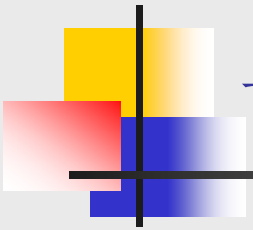
框架结构

- 用来实现分类——推理机制的数据结构。这是一种面向对象的数据结构。对象类型和对象属性；继承关系：属性的共享；对象类型和对象实例。



基于案例的推理

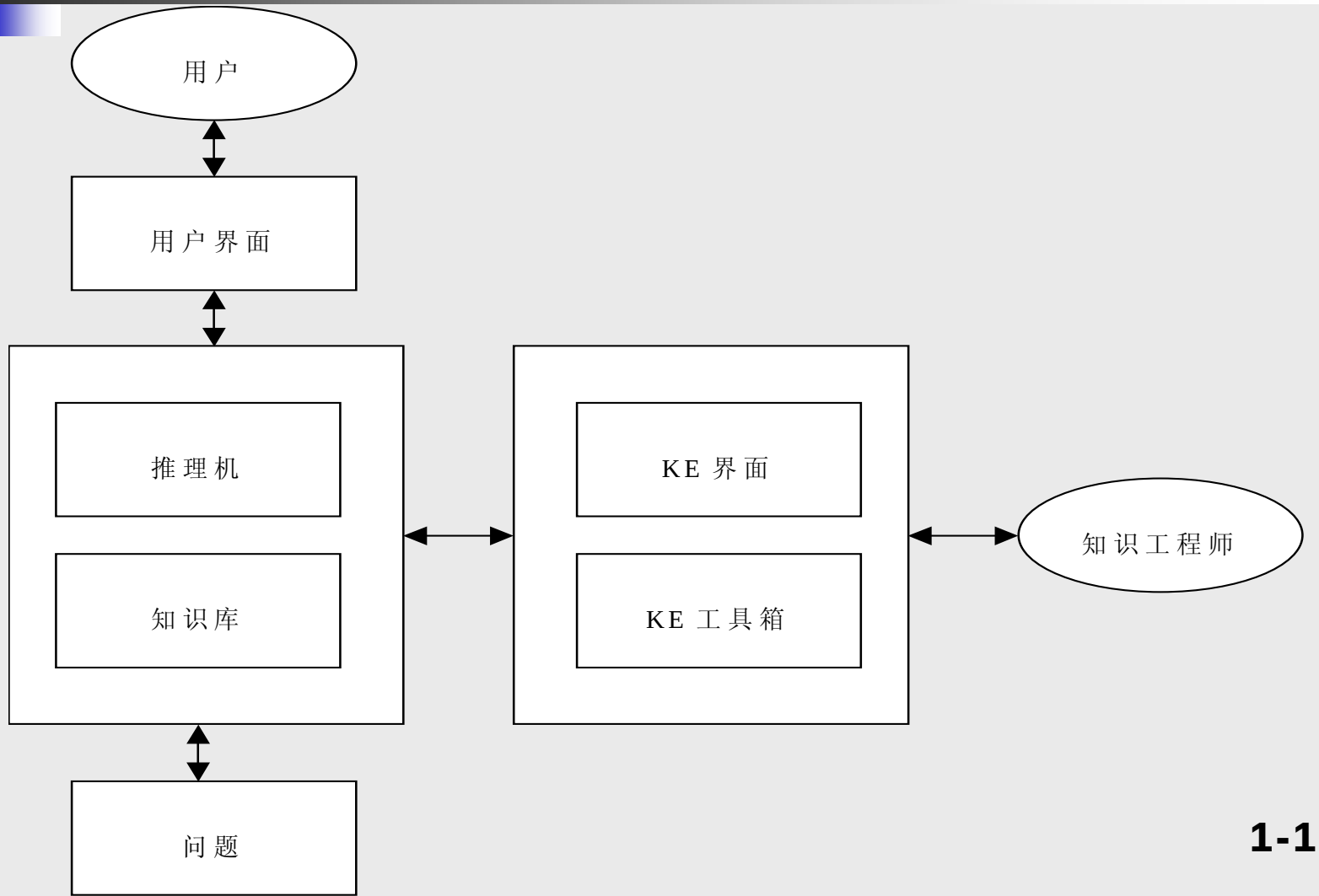
- 基于案例的推理运用相似性和经验知识进行推理和学习，解决问题。这种推理的思想是匹配，是在事实和案例库中的案例之间建立匹配。一旦能够找到可以匹配的案例，就对这个案例中采用的解决方案进行分析，根据当前情况修改这个解决方案生成适用于新问题的备选方案。通过测试或者仿真测试这个备选方案，看它是否成功，如果不成功则重复进行，如果成功可以将目前这个新方案加入案例库中。



专家系统的结构

- 专家系统由用户界面，推理机和知识库三大部分组成。知识库和推理机——尤其是知识库——是利用专门的知识工程方法，由专人（知识工程师）利用专用工具生成的，这些工具就是所谓的知识采集工具。
- 对于专家系统而言，知识库部分是它最具有特性的部分。

专家系统结构简图

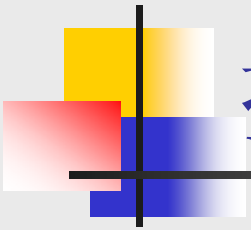


专家系统的用户界面

- 专家系统用户界面的开发相对独立于专家系统的设计开发。和决策支持系统一样，在专家系统中用户界面的相对重要性非常值得系统的设计者和开发者关注。为了实现一个比较好的专家系统用户界面需要大量的需求分析、设计和编码工作。
- 专家系统用户界面设计的焦点也和决策支持系统一样，在于“人性化设计”，让系统的输入简洁方便，同时具备必要的灵活性，能够减缓用户的疲劳并且尽可能减少用户出现输入错误的机会；让系统的输出直观，便于用户的理解和应用，减少用户的错误判断和给出的信息的二义性。

专家系统的知识库

- 这里包含有在特殊领域中应用的知识，而不是比较简单的数据。这些知识是从领域专家那里收集来的，包括对领域中各种对象的描述、领域中对象之间关联的描述、解决问题的操作、约束条件的描述、启发性知识、不确定性等等方面的知识。
- 专家系统解决问题的能力基本上依赖它的知识库中贮存的知识的完善程度和准确程度。如果知识库中包含的知识不完善、不一致或者不准确，那么无论专家系统的其他模块能够运行的多么正确，多么有效率，根据这些知识得到的最终结果只能够是对问题的不完善甚至是不正确的解。



推理机

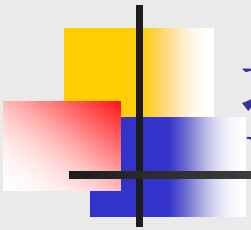
- 专家系统的推理机部分是进行推理工作的核心部件，而知识库则是推理工作的基础。推理机根据输入的问题，利用存贮在知识库中的知识进行推理，得到有关问题的结论。
- 推理机工作依据的是基于事实和规则的演绎推理，必要的时候也要进行基于概率的推理或者基于模糊匹配的推理。

推理机的基本工作原理

- 推理机进行推理的基本过程是控制循环。一个推理的控制循环可以分成三步：
  - 利用已知或者给定的事实寻找能够与之匹配的推理规则；
  - “执行”找到的匹配规则得到推理结果；
  - 将获得的推理结果加到“工作存贮区”中，作为下一次推理的出发点。

推理机工作依据的基本推理规则

- 如果有A，并且有A——〉B，
- 　　则有B。
- 或者是类似于假设检验的反向演绎推理规则：
- 如果有A——〉B，并且有非B，
- 　　则有非A。



推理机的基本工作方法

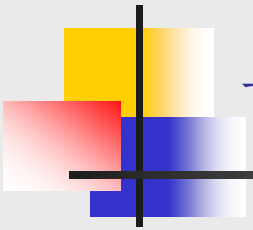
- 推理链法
- 分解法

推理链法

- 一个推理链条就是一个按照回归形式组织起来的、不断重复的控制循环：当前控制循环获得的推理结论作为下一个推理的控制循环的起点事实，直到无法进行进一步推理或者得到了期望的结果为止。
- 根据推理是从原因指向结果还是从结果指向原因，可以有两种推理链：正向推理和反向推理。
- 正向推理过程描述：推理机首先初始化临时推理工作空间，然后根据当前已知事实进入控制循环；一旦找到了匹配规则，通过执行它将相应的结论加入到临时推理工作空间中去，重复上述步骤直到根据准则推理可以结束为止。
- 因为这种推理是利用推理规则从起始事实向目标事实的移动，因此又被称为（事实）数据驱动的推理。

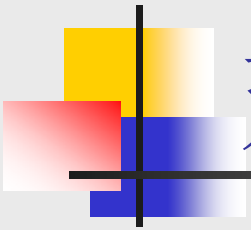
专家系统的临时推理工作空间：黑板

- 专家系统的临时推理工作空间是专家系统执行各种操作的时候记录临时结果的专用空间。这一空间还可以用来在专家系统运行过程中和用户进行交流，获得推理需要的额外信息或者向用户给出推理的中间结果，所以这块区域有一个特殊的名称：黑板。
- 利用黑板可以很方便地实现交互式的辅助专家系统，或者利用专家系统进行辅导式的决策咨询。



专家系统的设计和开发

- 如何低成本、高效率地开发专家系统一直都是专家系统应用中的难题。幸运的是这个问题目前已经有了成熟的答案，那就是专家系统生成器或者专家系统开发环境。
- 不同的专家系统有相同的共性：推理机部分完全可以通用。但是知识库和用户界面无法通用，这是因为对于不同的专门领域，解决问题需要的知识显然是不同的。
- 对于知识库的开发，当前的专家系统开发环境提供了专门的工具：知识工程工具来帮助进行知识库的建立，知识采集和知识录入等等工作。
- 比较麻烦的是专家系统用户界面的开发。对于这个问题目前都没有什么规范化、系统的化的应对策略，绝大部分工作要由系统的设计和开发人员利用传统的方法，结合专家系统开发的特殊知识完成，很大程度上算是“艺术”。

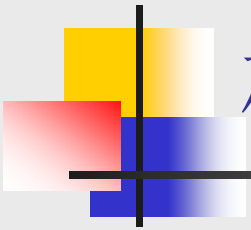


建立专家系统的目的

- 专家系统是针对特定目标的专门系统，因此必须在设计和开发专家系统之前将专家系统的应用领域和能力界限确定清楚。

专家系统能够完成的任务的大致类型

- 解释：利用已知的信息推断实际情况的含义和归属
- 预测：根据给定的情况和历史信息，推断将来最有可能发生的结果
- 诊断：根据观察的结果和对这些信息的分析，推断错误产生的位置和原因
- 设计：为满足特定的需要制定项目的计划
- 监控：将当前实际情况的信息和期望结果相比较
- 控制：根据计划和反馈信息管理整个系统的行为
- 指导：根据实际情况和反馈对用户当前行为做出判断，并通过给出建议指导用户行为



确定专家系统在组织中的位置

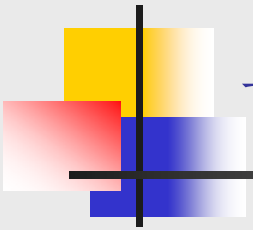
- 在组织中很多地方都需要专家系统。理论上任何人类专家任务繁重，或者人类专家能力不足，或者聘请专家成本高昂，或者很难聘请到相应专家的位置和情形，都可以考虑通过建立专家系统去解决或者帮助解决组织的问题。

专家系统设计和开发之前必须要进行的准备活动：寻找专家

- 专家系统设计和开发的成功与否，和专家系统拥有的知识的数量和正确性直接相关。因此开发设计专家系统之前必须能够找到一批可以提供足够数量并且正确和权威的专业领域知识的专家，没有这些专家就根本不可能建立专家系统。
- 说服专家投入大量的时间和精力。不仅如此，设计和开发专家系统还需要这些专家们投入大量的时间和精力。为了保证开发的成功，他们必须乐于参与设计开发的整个过程，付出大量的时间和精力。因此有必要让专家们了解相应的专家系统对于组织的重要性，以及他们的投入对于专家系统开发成功的重要性。
- 真正获得专家们的支持。专家系统应该定位于增强专家的作用，而不是取代专家。因为如果是后者的话，专家对专家系统所持的敌视态度，有可能会导致系统开发的失败。

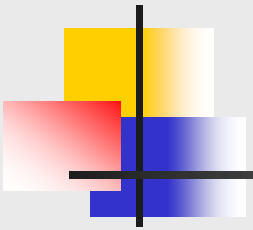
专家系统的收益和缺陷

- 收益可以从下面几个方面考虑：
- 增加组织制定决策的及时性。
- 提高组织内专家的生产效率。
- 提高决策的一致性。
- 提高人们对于决策过程的理解。
- 能够格式化、系统化地保存知识。

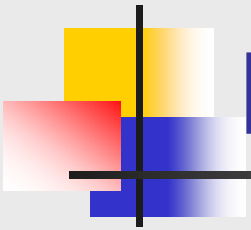


专家系统的局限性

- 解决问题需要的知识并非总是能够得到。
- 专家系统在利用常识方面能力有限。
- 很多专业知识和启发式规则是很难提炼并且转化成为知识库的规则或者专家系统的代码的。
- 只能够解决狭小的专业领域中的问题，对于解决超出了知识库领域的问题，专家系统几乎无能为力。
- 专家系统无法帮助用户消除感知的局限和各种偏见。
- 仍然不具备人类专家所具有的学习能力、创造力和对新环境、新问题的适应能力。
- 另外由于知识和规则的不完备性，专家系统常常犯错误。只有技术上的改进和创新才能够从根本上改变这一点。



- DSS的设计和构建也是一个具有挑战性的问题，这个问题不同于组织使用的任何其他应用系统的设计和开发。如何尽可能高效率、低成本地开发满足客户要求的DSS到今天为止都没有能够完美地解决。虽然对于DSS的设计开发方法的研究取得了很大进展，但是目前对于DSS的设计和构建问题的共识是：并不存在设计和构建DSS的最好途径；或者到目前我们还没能够找到这样的途径。

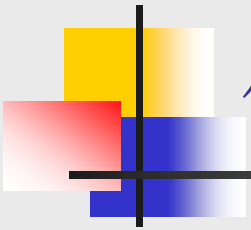


DSS的开发方法

- 有两种开发DSS的方法：A、编制一个用户定制化的DSS；B、使用DSS生成器生成需要的DSS。
- 在具体选择哪一种方法进行开发一般根据DSS需要解决的问题的要求和开发环境而确定。在复杂的DSS系统开发中非常有可能在不同阶段同时交错使用这两种开发方法。

编制一个用户定制化的DSS

- 这是最标准也是最普通的DSS开发方法。具体做法是采用一种通用编程语言（PASCAL，Delphi，VC）进行系统设计开发。虽然当今的编程语言以及开发平台提供的功能能够大大减少一般软件系统的开发工作量和开发时间，但是它们并不能够为DSS的开发提供什么直接的帮助。



使用DSS生成器

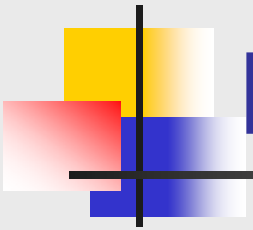
- 这是近期DSS开发的一个显著进步。DSS生成器是一种应用软件，它提供的功能能够直接减少DSS开发需要的工作量，节省开发时间。例如，电子数据表格就是一种最为常见也最为简单的DSS生成器。
- 虽然利用DSS生成器开发DSS能够大大提高开发效率，但是它却限制了开发的灵活度以及最终获得的DSS的功能和复杂程度。

DSS系统的分析与设计方法：生命周期法

- 这种方法是标准的软件开发方法，它将软件从开始设计到最终被淘汰的整个过程看作是软件的生命周期，从动态的角度将软件开发的整个过程分割成一系列递归阶段，每个阶段都有自己独特的输入、处理和输出，实现特定的目的。
- 在每一个阶段，开发工作都获得一个关于最终系统的模型，以及相关的大量文档。这些模型都是关于同一个软件系统的需求信息，只是在形式上将越来越规范化、精细化和结构化。

在问题定义阶段

- 系统分析人员评价开发问题的性质，并得出一份描述这些问题和问题环境的文档；在可行性分析阶段，通过考虑当前可用的技术水平、人力财力资源等约束，评价能将来开发出来的系统能否解决这些提出的问题；在系统分析阶段系统分析人员要收集资料，将需求精细化和文档化；在设计阶段将为系统构建数据流图等需求模型，对系统进行概念建模；在编程阶段通过开发能够满足用户需求的硬件和软件实现系统；测试阶段通过一系列测试分析，检验系统是否满足需要，以及是否需要反复和改进；实施阶段要进行最后的安装调试工作，以及进行用户培训；维护阶段将持续系统的整个剩余生命周期，包括改善系统，解决随时出现的问题，提高系统性能和升级系统。

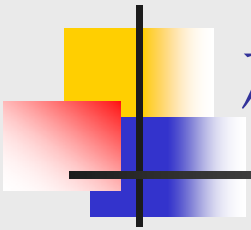


DSS的分析、设计和开发过程

- 确定是否需要DSS
- 确定DSS的目标和可用资源
- 系统分析
- 系统设计
- 系统构建
- 系统实施
- 逐步改善

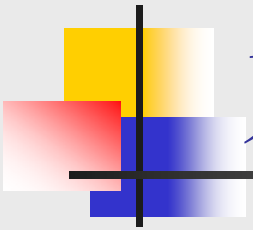
确定是否需要DSS

- 在组织中很多地方都可以应用DSS。对于如何发现在组织的什么地方需要应用DSS要依靠组织的参与者和管理者的帮助，由他们直接提出相关的要求。这种提出要求的过程也就是认清需要DSS支持解决的问题的过程，即按照规则识别和定义问题的过程。这可以通过对“问题症状”的研究来实现。



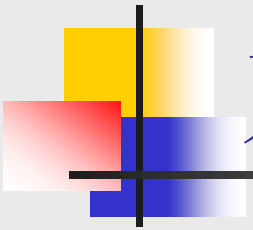
确定DSS的目标和可用资源

- 一旦确定了需要DSS解决的问题，那么下一步必须完成的工作就是描述用DSS实现的目标是什么：DSS提供什么样的决策支持以及当前可用的，能够用来实现这些要求的资源有哪些。
- 这个阶段得到的结论可以用来规划和DSS系统开发配套的知识工程的相关工作：确定需要采集的知识的范围和深度，安排知识工程计划；以及确定DSS设计和开发的进一步计划，建立用来衡量DSS是否实现预期目的的标准。这个标准一定要量化，具备可比性。



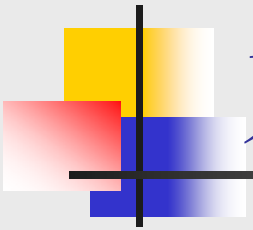
系统分析

- 和普通软件的生命周期法类似，在这个阶段，要详细确定DSS的需求，这些需求具体而言可以分成三类：功能需求、界面需求和协调需求。
  - 功能需求。这些需求主要包括获取、存贮和生成对解决特定问题有用的知识。
  - 界面需求。这是关于DSS界面交互功能的定义。
  - 协调需求。所谓的协调需求就是对决策动作发生的时间和次序的安排。



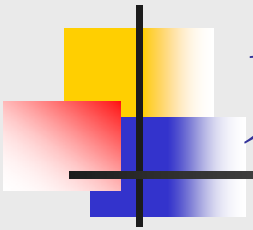
系统设计

- 在这个阶段要确定DSS开发和运行所使用的硬件和软件，包括开发平台和运行平台。其中最为重要的就是选择一套开发DSS的开发工具。



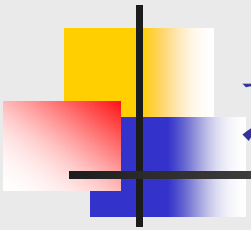
系统构建

- 在这个阶段设计人员采用一些专业方法，通过测试和用户反馈不断地对系统的需求和模型进行调整和改进。本过程在整个DSS的开发阶段延续，常常导致DSS设计中的显著变化。系统构建可能需要像以前的阶段回溯，这种反复是经常的也是必需的。通过系统构建过程中的反复可以深入发掘需求：很多没有被明确提出的需求就是在这个过程中被提出和收集的。
- 这是DSS设计和开发的主要阶段，占用了整个开发过程的大部分时间，以及开发人员的大部分精力。



系统实施

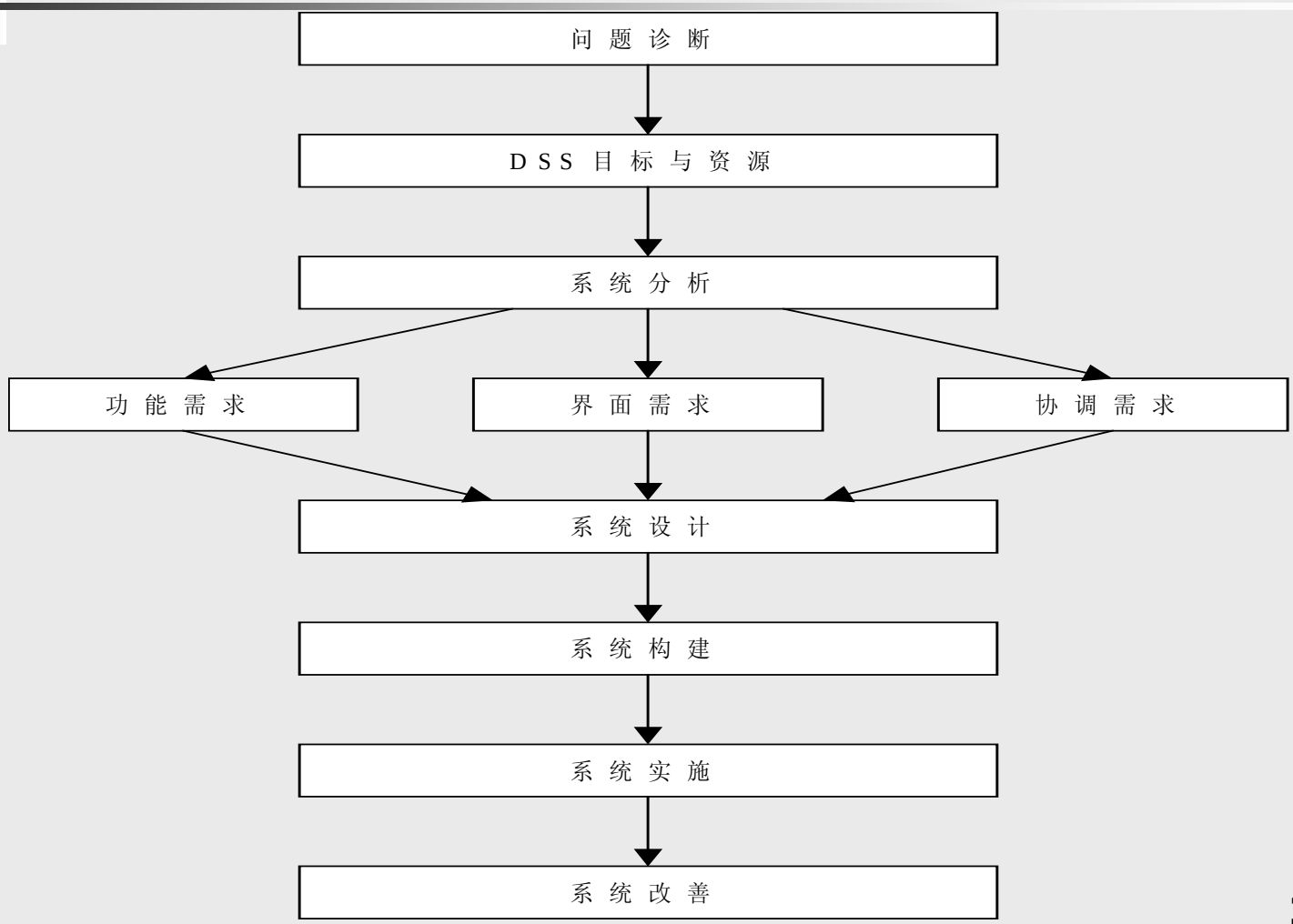
- 这个阶段的目标是测试、评估并且配置一个文档齐全、功能完善的DSS。这个阶段还要培训DSS的客户群，让他们了解系统的结构、功能和使用方法。



逐步改善

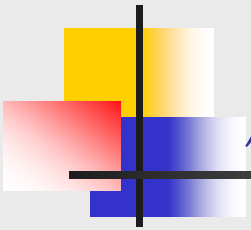
- DSS投入运行之后的逐步改善要一直持续下去，直到DSS系统被放弃为止。这种逐步改善可能是没有尽头的：从实施和维护老的DSS中获得的各种经验完全可以应用在新的DSS开发上。
- 只要不被中断，DSS的开发完全可以是一个连续的、进化的生命过程。随着不断的升级，DSS的能力会不断增加，最终也许会进化成完全不同的、稳定而且全面的DSS系统。

典型的DSS开发过程如下



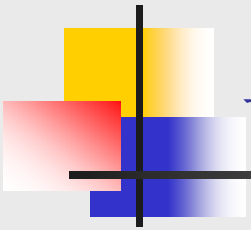
生命周期法并不是很适合DSS的开发

- 生命周期法是一种高度结构化的软件开发策略，并且也被认为是最有效的软件开发策略。
- 这是一种标准的自顶向下的分析、设计方法：首先需要确定系统的确切需求和特点，然后通过不断细化的系统设计和构建过程，发掘系统需求，最终得到完善和全面的系统需求。
- 应用生命周期法的时候，都假设在设计阶段开始的时候就能够充分了解并且识别系统需要解决的问题以及问题的环境，这种要求对DSS的设计开发工作而言并不能够成立。
- 由于DSS要解决的问题是半结构化问题，所以问题本身的特性，问题环境的描述等等设计和开发DSS系统需要事先了解的关键问题只有在开发过程中才能够逐渐确定，因此从这一点上说，和生命周期法是相矛盾的。



原型化开发方法

- 原型化开发方法是一种反复的和进化的开发方法。
- 原型是最终的软件系统的雏形。它能够运行，具备按照需求分析所描述的系统的功能。原型的功能在于为需求交互提供一种可靠和有效的手段，通过原型，开发人员和用户能够快速找出需求中的误解和不足。
- 这种方法强调在设计和构建系统的过程中通过开发原型进行需求交互，以次逐渐了解和掌握系统的要求，因此比较适合DSS的设计和开发。
- 原型化开发方法利用原型帮助开发人员进行需求分析，同时将用户在设计开发中的作用的从被动参与改变为直接参与。



丢弃型的原型开发

- 这种开发使用丢弃型原型，即原型只起到示例的作用，完成了需求交互的示范工具作用之后就被马上抛弃。这里的原型只有辅助需求交互的作用。这种开发方法可以节省开发时间和经费。

反复型的原型开发

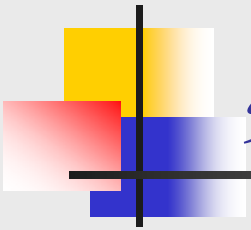
- 在这种开发方法中，原型在用来进行需求交互之后也不会被抛弃，而要在需求交互之后对原型进行改进甚至是重新开发，直到它最后完全满足DSS用户的需求，进化成为开发最终产生的结果。

原型化开发方法的优点和局限

- 对于DSS的开发，原型化开发方法似乎更加有效。这种方法能够显著减少设计开发花费的时间和费用；能够让用户给出关于系统功能需求的及时和明确的反馈；以及促进用户对于DSS在组织中的定位和在功能上的理解。
- 原型化开发方法的局限表现在：对比生命周期法而言，对整个系统的开发文档的重视度不够，不太容易关注开发文档的细节；因为原型的突出性和鲜明性，导致将主要的注意力集中在DSS的功能和易用型上，忽视了系统同样非常重要的其他方面：维护的难易性、文档的准确详细性以及是否符合软件开发规范等方面。
- 不过这些问题并不是不可克服的。通过完善和规范化原型化开发方法可以基本解决它们。

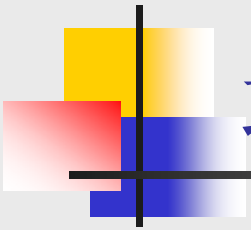
DSS系统的开发人员

- 从构成上来看，决策支持系统的开发人员是多种多样的。这里面既有经验丰富的专业系统分析人员；也有第一次参与开发的新手。即使是所谓的专业人士也包括了很多类型：既有专业的DSS应用系统开发设计师，也有那些没有任何DSS开发设计经验的软件开发专业人士；而那些第一次参与开发的新手通常都是DSS的用户。对所有这些参与DSS系统开发的开发人员来说，经历一次这样的开发过程将会是非常有益的经验。



必须具备的能力

- 参与DSS开发的人员，不论其在开发中的角色、行使的功能如何都必须具备一些关键的能力。
  - 理解问题领域的知识。
    - 从这个意义上说，DSS的用户，以及专门负责某个领域的DSS开发的专业开发人员具有相当的优势。
  - 理解具体用户的需求。
  - 掌握可行的开发技术。
  - 采集和表达需求的能力。



最终用户开发DSS

- 如果那些在组织中直接使用DSS的用户，他们从自身的实践中发现了对DSS的需求，并且掌握有开发DSS的相关技术的话，这种DSS开发方法倒是一种非常好的选择。

最终用户开发DSS的优势

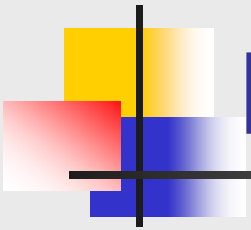
- 最终用户一般都非常了解和熟悉DSS的环境和需要解决的问题，并且掌握相关领域的专业知识，因此他们对于需求是了解的。
- 因此，最终用户开发DSS可以节省甚至取消DSS设计和开发阶段中耗时耗力的需求收集、规范需求和需求建模阶段；同时还可以大大减少系统开发中因为误解需求而产生的种种问题，缩短DSS系统投入应用需要的磨合期的长度；因此减少DSS系统开发需要的时间以及相应的成本。

最终用户开发的风险

- 一般最终用户都不是专业的软件系统设计和开发人员，他们一般都缺乏软件系统开发方面系统和正规的训练，而且开发DSS的经验、能力参差不齐。因此最终用户开发面临的最大风险就是开发出来的DSS系统缺乏必要的质量保证。因为最终用户在开发的过程中常常会有意无意地回避开发中的控制机制和测试过程，导致开发过程混乱，最终产品质量低下，可能包含严重的甚至是致命的缺陷，甚至开发的完全失败。
- 同样的原因，最终用户开发的另外一个常见风险是无法生成质量可靠的系统文档，这对于系统的维护和升级非常不利。
- 由于最终用户一般对于计算机系统整体的不熟悉或者考虑不周，可能导致最终产品缺乏安全性。
- 这些风险并不是不可控制的。通过提供合适的开发工具、开发过程框架、组织上的支持（例如建立企业中的信息中心）或者基本培训，可以大大减少最终用户开发产生的风险。

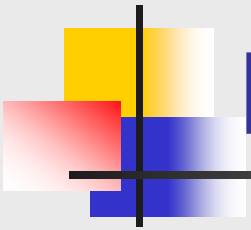
DSS系统的开发工具

- 选择合适的DSS开发工具也是一个重要决策。
- 一般而言，有三类DSS开发工具可以选择：
  - DSS基本开发工具
  - DSS生成器
  - 专门的DSS应用系统



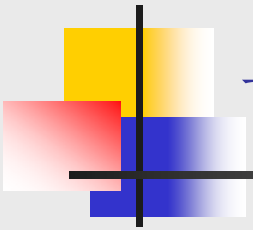
DSS基本开发工具

- 这些工具属于开发DSS的最低层技术，不过它们能够应用在最为广泛的目的中，不仅仅包括DSS开发。但是应用这些工具需要的相应计算机知识和系统开发知识也是最多的。



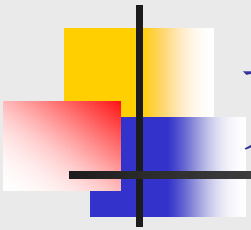
DSS生成器

- 这是一套帮助迅速方便地建立专用DSS的硬件和软件，它为DSS开发提供的最大好处就是方便。



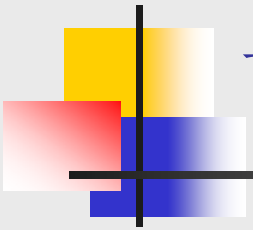
专门的DSS应用系统

- 如果问题不是非常特殊的话，那么使用专门开发的DSS应用系统就可以解决问题。



如何选择

- 面对特定的DSS开发要求，究竟应该如何选择：是开发特殊的DSS系统，还是应用当前就可以在市场上获得的专用DSS系统；是利用基本开发工具从头开发还是应用DSS生成器？这是一个非常重要的决策，也是一个非常困难的决策。



可以考虑如下依据来进行判断

- 问题的特殊程度
- 费用
- 可选择系统的通用性
- 销售商提供的支持服务的质量和可得性
- 系统应用的要求，例如硬件要求