使用 AI 助手开发用于形式化规约模式的受控自然语言

Natalia Garanina 信息物理系统实验室 自动化与电测量研究所 新西伯利亚，俄罗斯
natta.garanina@gmail.com

Vladimir Zyubin 信息物理系统实验室 自动化与电测量研究所 新西伯利亚，俄罗斯
zyubin@iae.nsk.su

Igor Anureev 信息物理系统实验室 自动化与电测量研究所 新西伯利亚，俄罗斯
anureev@gmail.com

摘要 — 我们借助 AI 助手，开发了一种系统构建需求受控自然语言的方法，该方法基于包含逻辑属性的形式化规约模式。该方法包括三个阶段：1）编写一个通用的自然语言需求模式，该模式利用形式化规约模板的所有属性；2）利用 AI 助手生成自然语言需求模式语料库，并通过对属性进行部分求值来缩减该语料库（所开发的提示使用通用模板、属性定义以及需求模式的特定形式语义）；3）基于对所得模式语法结构的分析，形式化受控自然语言的句法。该方法已针对事件驱动的时序需求进行了测试。

关键词—受控自然语言、需求、形式语义、AI 助手、模式

I. 引言

开发一种用于描述事件时序需求的受控自然语言（CNL），是软件需求工程领域亟待解决的研究挑战。在当今世界，计算机系统在金融、医疗保健、交通运输和工业自动化等行业中日益重要，确保对系统事件及其时序特征进行准确且易于理解的规范描述，发挥着至关重要的作用。根据 [1]，软件开发中的大多数错误都发生在需求规范阶段，因为工程师、分析师和其他利益相关者会误解需求。初始需求中的任何错误都意味着需要在后续进行修正，这既耗费时间，又会增加成本 [2,3]。

我们开发了一种方法，利用 AI 助手，根据包含逻辑属性的形式化规范模式来系统地构造受控自然语言需求。该方法包括三个阶段：1）用自然语言编写通用需求模式，该模式使用形式化规范模式的所有属性；2）利用 AI 助手生成自然语言需求模式语料库，并通过对属性进行部分求值来简化语料库（所开发的提示使用通用模式、属性求值结果以及需求模式的特定形式语义）；3）基于对所得模式语法结构的分析，将由此得到的受控自然语言的句法形式化。

我们在事件时序需求语言 EDTL [4] 上测试了所提出的方法。

我们的方法适用于这样的模式：其自然语言解释依赖于恒定的属性值，而这些属性值可简化公式。与先开发需求语言、再定义其形式语义的其他方法不同，例如 EARS [5] 和 Rimay [6]，这种语言从设计之初就具有形式语义。还有一些已具备形式语义的需求语言，它们将若干逻辑公式与自然语言表达相关联，但这类语言相当有限，表达能力不足 [7, 8]。在其他情况下，例如 SysReq[9]，，需求语言虽具有较强的表达能力，但其形式语义难以表达且不够简洁，因此也难以验证。

因此，受控自然语言可以通过多种方式开发：遵循形式语义进行构建，或者反过来，根据领域工程师的需求（这些需求并不总是经过良好形式化）进行构建，然后将逻辑公式与由此产生的自然语言构式相匹配。目前，既有语义复杂且具有较强表达能力的受控自然语言（SysReq [9], BraceAssertion [10],PROPAS [11]），也有语义简单、表达能力较弱的受控自然语言 [7,8, 12-15]。

我们建议构建受控自然语言所采用的需求规格说明格式包含三种简洁的表示法：表格表示法（需求属性的逻辑值）、形式表示法（需求语义公式）和语言表示法（自然语言短语）。这些表示法简洁且易于理解，同时通过多样的属性值组合确保需求具有足够的表达能力。除了我们开发的 EDTL 之外，德国公司 BTC 开发的 SUP 需求表示格式 [16-18] 也具备类似的特性。

我们系统化的受控自然语言开发技术具有以下优势。单一的初始自然语言模式以及三值布尔属性（取值为真、假和变量）的可用性，使我们能够从紧凑的初始表示生成范围广泛的句子。此外，单一的初始自然语言模式便于与人工智能助手协作，也就是编写提示模式，并确保生成的短语具有一致的语法结构，从而有助于

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

国际

语法构建。此外，每个生成的短语在设计上都具有形式语义。

II. 预备知识

受控自然语言（CNL）是某种自然语言的一个子集，其句法和词汇经过明确（限制性）定义，因此其句子可以无歧义地转换为某种形式逻辑 [19]。典型的受控自然语言使用该语言语法和词汇的一个定义明确的子集，同时添加特定领域所需的术语。

我们考虑可使用随时间变化的变量值来描述的系统的形式行为规范，这些变量值定义了系统在不同时间点的状态。形式化需求规范模式是一组属性。每个属性以及整个属性集，都有一段直观、非正式的说明，用以描述其在系统需求规范中的用途；而它们的形式语义则是某种逻辑的公式。由于每个属性的形式语义都是一个逻辑公式，这些属性的逻辑值可以为真、假或 var——通常是依赖于系统状态、并由系统变量上的公式描述的变量值。下面我们给出 EDTL 和 SUP 形式化规约模式的示例。

A. 事件驱动时序逻辑（EDTL）

作为一组属性的模式。事件驱动时序逻辑（EDTL）是一种形式化描述需求的方法。这种记录需求的方法使用户能够通过事件以及与控制对象和环境的交互来描述控制系统的行为，并将控制系统和控制对象视为“黑箱”。

EDTL需求 R 是由六个 EDTL 属性组成的元组：R = (触发, 不变式, 最终, 延迟, 反应, 释放)。EDTL需求的属性是受特定时序关系约束的系统事件。

属性的非正式说明。示例。下面描述 EDTL 属性的非正式含义：

- 触发是需求的初始（触发）事件；
- 不变式是一个条件，从触发事件发生时起一直成立，直到发生释放或反应；
- 最终是一个发生在触发之后、并开始等待反应的事件；
- 延迟是一个事件，用于指定反应事件必须在最终事件发生后的哪个时间间隔内发生；
- 反应是需求所期望的事件；
- 释放会取消需求：若发生该事件，则无论其他需求属性对应的事件是否发生，都视为需求已满足。

干手机 EDTL需求示例：

如果烘干机下方有手且烘干机已开启，烘干机将持续运行。 **(1)**

在此需求中，触发条件是双手位于烘干机下方且烘干机正在运行。对触发的响应是烘干机继续运行。此需求必须始终得到满足，因此不会发生释放事件。触发后，烘干机继续运行不需要满足任何特殊条件，也没有任何时间约束，因此无需指定不变式、最终事件和延迟。

模式及其属性的形式语义。EDTL 模式的形式语义为线性时序逻辑公式 LTL[20]:

$$\boldsymbol{G}\,(trig \rightarrow ((inv \wedge (\neg\, fin))\ \boldsymbol{W}\ (rel \vee fin \wedge ((inv \wedge (\neg\, del))\ \boldsymbol{W}\ (rel \vee (inv \wedge rea))))))$$

在此公式中，命题陈述 trig、inv、fin、rel、del 和 rea 指定需求属性的逻辑值；根据系统变量的取值，它们可以是布尔常量或命题公式。

需求 (1) 的属性值列于表1：

表 I. EDTL 需求 (1) 的属性值

触发	释放	最终	延迟	不变式	反应
H ∧ D	false	*true*	*true*	*true*	*D*

这里，布尔变量 H 表示烘干机下方是否有手，D 表示烘干机是否开启。

我们已证明，EDTL 模式符合形式化规范模式的标准，因此可应用我们构建受控自然语言的方法。

B. 简化通用 SUP 模式

作为一组属性的模式。BTC 的简化通用模式（SUP）是一个包含 15 个参数的单一模式。大多数参数都有默认值。BTC Embedded 平台使用图形化编辑器进行形式化，使工程师只需填写必需的参数；具有默认值的参数会对用户隐藏。SUP 定义了触发与动作之间的关系。通常，触发是被规范描述的系统可在其端口上观察到的某种行为，而动作则是该系统必须执行以作出响应的某种行为。SUP 有三种解释：进展（触发之后发生动作）、不变式（触发与动作同时发生）和顺序（动作之前总是先发生触发）。本文采用最常见的解释——从观察者的角度描述的进展。观察者是一个与系统并行运行的组件，用于监视端口，并在指定行为遭到违反时生成错误信号。

请在此处注明适用的资助机构。若无，请删除此文本框。请在此处注明适用的资助机构。若无，请删除此文本框。

属性的非正式描述。示例。SUP 模式的属性可作如下非正式描述:

触发阶段。触发阶段由触发开始事件（TSE）、触发条件（TC）、触发结束事件（TEE）和触发退出条件（TEC）定义，并受两个时间约束限制：触发最小值（Min）和触发最大值（TMax）。从技术上说，事件与条件相似。使用“事件”一词是为了强调该条件标志着某个阶段的开始或结束。如果 TEE 在 TSE 之后的 [TMin, TMax] 时间范围内发生，且 TC 在此时间间隔内得到满足，则认为 SUP 触发已成功捕获。如果 TEC 得到满足，则观测周期终止。

动作阶段。动作阶段与开始阶段类似，由动作开始事件（ASE）、动作条件（AC）、动作结束事件（AEE）、动作退出条件（AEC）、动作最小值（AMin）和动作最大值（AMax）参数定义。AEE 必须在 ASE 之后的 [AMin, AMax] 时间范围内发生，且 AC 必须在此期间发生。否则，观测失败。如果 AEC 发生，则当前观测周期中止，并开始新的观测周期。如果成功观测到 AEE，则当前观测周期成功，并开始新的观测周期。

局部响应区域。由两个参数组成的局部响应区域 [LMin, LMax], , 用于限制 TEE 与 ASE 之间的时间间隔；如果 ASE 观测得过早或过晚，则观测失败。

SUP监视器是一种状态机，因此每个观测周期只记录首次触发。默认情况下，TSE = TC = TEE、ASE = AC = AEE、TEC = AEC = false、AMin = AMax = TMin = TMax = LMin = Lmax = 0。在以下示例中，我们假设 SUP 使用默认值，并且仅指定必需的值。

下面给出两个以文本形式和使用 SUP 表示的需求示例（表 2）：

(A1) inp_1 每 35 ms 为真一次。

(A2) inp_1 每隔 30-40 ms 为真一次。

表 II。 SUP 需求的属性值

ID	***Trig*** TSE = ***TC*** = ***TEE***	*[*LMin, LMax*]*	动作			
			ASE	***AC***	***AEE***	*[*AMin, Amax*]*
A1	*true*	[0, 0]	*true*	¬输入1	inp_1	[35毫秒, 35 毫秒]
A2	*true*	[0, 0]	*true*	¬输入1	inp_1	[30毫秒, 40 毫秒]

该模式及其属性的形式语义。作者将 SUP 模式的语义定义为计数自动机。不过，也可以像 EDTL 模式那样，将其定义为线性时序逻辑公式，但该公式涉及无界整数变量的命题陈述。此外，在许多情况下，包括示例案例，标准线性时序逻辑就已足够。虽然为 SUP 定义线性时序逻辑语义超出了本文的讨论范围，但在构建受控自然语言时，该模式的非正式定义意味着其属性也取真值 true 或 false，这一点很重要。

作为触发和动作执行条件或其开始和结束标记（TSE、TC、TEE、ASE、AC、AEE、TEC、AEC）的属性显然是布尔值;

作为时间区间的属性解释如下:

- 局部响应区域 [Lmin, Lmax] 是对响应（用 EDTL 术语来说是最终事件）发生时间的约束，因此，区间 [0, 0] 对应 false 值——响应必须立即发生；而区间 [0, ∞) 对应 true 值——响应发生时间不受约束。所有其他选项都对应布尔语句 LS = (t ⩾ Lmin ∧ t ⩽ Lmax 的时变值，该语句约束动作的开始时间。

- 开始／动作的最小／最大时间 [Tmin, Tmax] 和 [Amin, Amax] ——这些时间区间的解释方式类似。对于触发，区间 [0, 0] (false) 表示触发永远不会发生，而区间 [0, ∞) (true) 表示触发可能在任何时间发生。对于动作，区间 [Amin, Amax] 类似于 EDTL 延迟属性：[0, 0] (false) 表示动作应立即执行，而区间 [0, ∞) (true) 则意味着动作的完成可能会无限期延迟。

因此，我们提出的用于构建受控自然语言的方法也可用于 SUP 模式。

III. 方法

使用形式化规约模式构建受控自然语言的系统方法如下。

A. 阶段1：基本模式

对于形式化需求规范模式，首先手动用自然语言构建一个基本规范模式。当其各个属性既非恒真也非恒假时，该模式由一个或多个自然语言句子组成，这些句子尽可能完整而简洁地描述需求的含义。这些句子必须包含用于替换所有模式属性值的槽位。理想情况下，应构建这些句子，使得将用自然语言表示的属性含义（即名称或短语）替换到槽位中后，所得结构在语法上是正确的。

该阶段的结果是一个包含所有需求属性槽位的自然语言句子模式。

B. 阶段2：派生模式

众所周知，如果属性取恒真值或恒假值，则与之对应的逻辑公式

其形式语义可以得到显著简化。需求公式的简化也会使其在自然语言中的表达更加简洁。在这一阶段，有必要明确：当基本规约模式中的某些属性为常量（恒真或恒假）时，该模式在自然语言中究竟会如何转换。为构造新的派生句型，我们使用基于大语言模型的 AI 助手，具体如下。

(1) 基本提示。我们定义一种文本查询模式（提示），为自然语言需求规范提供基本模式，并指定常量属性的值。此外，可以要求 AI 助手解释最终结果为何正确。这样，如果结果在语义上不够准确，就能据此确定如何改进向 AI 助手发出的查询。

基本提示可以单独使用，无需添加任何内容。不过，以下针对 AI 助手的指示可能有助于确保最终短语具有更高的“语言自然度”和语义正确性。

(2) 需求的形式语义。为了获得更精确的结果，可以在每个具体查询中，针对给定的常量属性值指定需求的形式语义。该形式语义通过将常量属性代入基本模式的语义公式计算得出。

(3) 对常量属性值直观含义的解释。这些解释会指导 AI 助手在逻辑属性为真或为假时如何构造最终句子。在这些情况下，常量属性的名称完全不会出现在最终句子中，但其具体含义仍会得到体现。实验部分介绍了一些示例。

此步骤的结果将是一个自然语言句型列表，其中包含用于属性的槽位。这些句型通常比原始句型更短。

C. 第3阶段：受控自然语言语法

借助 AI 助手构建与单个初始句型相匹配的句型列表后，通过使用大语言模型，生成的句型具有相似的语法结构。这简化了受控自然语言语法的构建。在此阶段，根据语法结构相似且包含属性槽位的句型列表构建语法。终结符是所构建句型中的词语，以及为不同属性指定的唯一槽位标识。示例见下一节。

IV. EDTL 实验

我们将以为 EDTL 模式（触发、不变式、最终、延迟、反应、释放）构建受控自然语言为例，说明我们的方法。在实验中，我们使用谷歌的 AI 模式作为 AI 助手。

A. 第 1 阶段：EDTL 的基本模式

EDTL 自然语言规约的基本句型如下：

在“触发”之后，“不变式”持续有效，直到“释放”或“反应”发生；并且“反应”必须在“完成”之后的“延迟”时间内发生。

B. 阶段2：EDTL 派生模式

此阶段会生成一组较短的派生短语，对应于规约模式的一个或多个属性取常量值的情况。为生成这组短语，我们使用了谷歌的 AI 助手。下面将通过 EDTL 示例说明第 2 阶段的步骤。

(1) 基本提示。为了使用 AI 助手，我们定义了一种基本提示模式，其中包含上一步开发的自然语言形式的基本形式化规约模式：

请用英语改写以下句子：“在“触发”之后，“不变式”在“释放”或“反应”发生之前一直有效，且“反应”必须在“最终”之后“延迟”时间内发生。”如果始终触发 = *、释放 = *、延迟 = *、最终 = *、反应 = *、不变式 = *。请解释改写后的句子为何正确。 **(1)**

此模式通过以各种组合将属性值常量代入“*”符号来指定。如果假定某个属性具有变量值，则会从提示的第二部分中将其删除。例如，对于组合：释放 = false、延迟 = true、最终 = true、不变式 = true（触发和反应为变量），具体提示可写为

用英语重新表述以下句子：“在‘触发’之后，‘不变式’一直有效，直到‘释放’或‘反应’发生；并且‘反应’必须在‘最终’之后的‘延迟’时间内发生。”如果始终释放 = *false*，延迟 = *true*，最终 = *true*，不变式 = true。

对于六个属性及其三个取值（true、false 和 variable），组合数量相当庞大：$36 = 729$。因此，在这一阶段，合理的做法是使用自动化替换所有可能的属性组合，或者利用模式语义分类的结果，尤其是 EDTL [20]。语义分类利用了这样一个事实：将常量属性值代入模式时，不同属性值组合所生成的形式语义会产生相同的公式。考虑到生成公式的时序结构，这一事实使得确定属性值的规范组合成为可能。对于 EDTL 模式，共有 32 种规范组合。例如，释放 = false、延迟 = true、最终 = true、不变式 = true 这一组合是规范组合，生成的语义为 G *(*触发 → 反应*)*；而释放 = false、延迟 = true、最终 = true、反应 = true 这一组合不是规范组合，因为它生成的语义与 G *(*触发 → 不变式 *)* 在时序上等价。

提示（1）是基础提示，无需进一步补充即可使用。不过，以下针对 AI 助手的提示可能会有所帮助。

(2) 需求的形式语义。给定常量属性值时，需求的形式语义使 AI 助手在构造最终短语时，不仅可以依赖原始模式，还可以依赖语义公式。除了提示 (1) 之外，我们用于指令的提示模式如下：

生成的句子必须对应以下线性时序逻辑公式“*”。 **(2)**

表 3 展示了仅使用提示 (1) 以及组合使用提示 (1) 和 (2) 的结果。

表 III. 使用提示 (1) 和 (1)+(2) 的自然语言模式

	属性	**LTL** 公式	提示 (1)	提示 (1)+(2)
1	释放 = false 删除 = true 完成 = true 无效化 = 真	G (触发 → rea)	在“触发”之后，the 条件应保持有效直到“反应”，其中 must 发生在指定的时间限制内。	在“触发”之后，“反应”必须发生。
2	完成 = 真	G（触发 →（（无效化 ∧（¬删除））W（释放 ∨（无效化 ∧ 反应））））	触发后，不变式应有效直到 任一“释放”or“反应”，其中 must 发生 于“延迟”。	在“触发”之后，“不变式”必须成立，且“延迟”不得发生直到 任一'释放' or '反应' 发生 .
3	释放 = false 删除 = true 反应 = true	G (触发 →((无效化 ∧ (¬完成)) W (完成 ∧ 无效化)))	在“触发”之后，不变式“应当有效直到 a 点必须发生 在.....之内 a 时间范围来自“最终”.	在“触发”之后，“不变式”必须保持有效，直到“最终”发生。

在这里，从第 1 行可以清楚地看出，只有使用提示（1）时，AI 助手才会“理解”常量属性不应出现在生成的短语中，而应直接使用它们的等价表达：例如，它用“条件”代替“不变式”，用“时间限制”代替“最终”和“延迟”。这种替换并不符合 EDTL需求的含义。将公式添加到提示中后，生成短语的语义就变得正确了。然而，使用提示（2）的缺点是，AI 助手不再关注原始自然语言短语，而只是将 LTL 公式“翻译”为自然语言，如第 2 行所示。当存在许多非常量属性时，这会导致繁琐且难以阅读的语法结构。因此，在下一次实验中排除了提示（2），但增加了对常量属性的解释。

(3) 属性名称使用限制。对于某些属性具有常量值的情况（在 EDTL 中，这些属性是延迟和最终），AI 助手会做出错误推断，并构造语义上不正确的语句。例如，如果 delay = false，它会认为时间限制无效，因此永远不会有响应（这一点从我们要求助手提供的解释中可以看出）。在这种情况下，明确常量属性值的含义会很有帮助，因此我们拟定了如下形式的提示。

请记住，如果不变式为真，那么该语句不依赖于不变式；最终 = true 表示最终时刻就是现在，最终 =false 表示最终时刻永远不会到来；延迟 =false 表示延迟时间无限长，延迟 =true 表示没有延迟；反应 = false 表示我们不等待反应，反应 = true 表示该语句不依赖于反应。 **(3)**

下方的表 4 展示了将提示 (1) 与提示词 (2) 和 (3) 组合使用的结果。

表 IV. 提示 (1)+(2) 和 (1)+(3) 的自然语言模式

	属性	**LTL** 公式	提示 (1)+(2)	提示 (1)+(3)
1	释放 = false 删除 = true 完成 = true 无效化 = true	G (触发 → rea)	之后 “触发”，“反应”必须发生 .	在“触发”之后，“反应”发生 now.
2	完成 = true	G (触发 →((无效化 ∧ (¬删除)) W (释放 ∨ (无效化 ∧ 反应))))	之后 “触发”，“不变式” must 保持且“延迟”不得发生 直到 任一‘释放’ or ‘反应’ 发生。	在“触发”之后，“不变式‘ is 有效 直到 任一释放 or 反应，and 反应 occurs within 延迟 from now.
3	释放 = false 删除 = true 反应 = true	G (触发 →((无效化 ∧ (¬完成)) W (完成 ∧ 无效化)))	在“触发”之后，“不变式”必须保持，直到“最终”发生	在“触发‘ 之后，“不变式‘是永远有效的。

我们发现，使用提示词 (1) 和 (3) 时，AI 助手在大多数情况下能够构造出语义正确、语法通顺且简洁明了的短语。然而，某些属性值组合（尤其是带有常量

国际

反应值的组合）会使生成的短语具有更宽泛的语义。这是因为这类属性值组合的直观含义变得模糊不清，只有根据语义公式才能理解。我们认为，这类 EDTL需求对人类来说也难以理解，因此将相应模式纳入受控自然语言毫无意义。

总之，结合提示词 (1) 和 (3)，无论从句法还是形式语义来看，都能更好地构建受控自然语言语法。这种组合往往能够识别出一些需求属性值组合，而这些组合并不符合人们对属性值的直观理解。这就无需在用自然语言描述需求时使用这些组合，而是在以表格形式规定需求时提醒工程师注意潜在错误。

C. 第3阶段：EDTL 的受控自然语言语法

我们根据上一阶段通过提示词 (1)+(3) 得到的短语模式列表，介绍 EDTL 受控自然语言的部分语法。

需求：= 在 <触发>后， <主体_触发> | <不变式> 有效 <主体_无效化>

<主体_触发>： = <反应> 发生 <条件_反应> | <不变式> 有效，直到 **<条件_无效式>** | **...**

<条件_反应> := 现在| 从<最终> | 在<延迟> 从<最终> | ...

V. 结论

我们开发了一种方法，利用 AI 助手，根据形式化规范模式系统地构建受控自然语言（CNL）需求。该方法包括三个阶段。第一阶段是定义一种基本的自然语言需求模式，该模式使用形式化规约模式的所有属性。第二阶段是利用 AI 助手构建具有特定属性的派生自然语言需求模式。这一阶段利用需求模式的形式语义。第三阶段则使用所得模式构建受控自然语言语法。需要指出的是，所采用的大语言模型生成方法简化了对这些模式语法结构的分析。

自动化第二和第三阶段将拓展受控自然语言的构建能力，使其适用于任意形式化需求规范模式。

致谢

本研究得到俄罗斯教育与科学部项目 125022803031-1.. 的资助。

参考文献

[1] B. W. Boehm, 《软件工程经济学》, IEEE软件工程汇刊, 第1期, 第4–21页, 1984年。[2] Meziane F. 从UML类图生成自然语言规格说明 / F. Meziane 等 // 需求工程。– 2008。– 第13卷。– 第1期。– 第1-32页。– doi:10.1007/s00766-007-0054-0。[3] Mavin A. 倾听经验（应用EARS的8条经验教训） / A. Mavin 等。– 2016。– 第276-282页。– doi:10.1109/re.2016.38。

[4] V. Zyubin、I. Anureev、N. Garanina、S. Staroletov、A. Rozov和T. Liakh, 《用于控制软件需求规格说明的事件驱动时序逻辑模式》, 载于《软件工程基础国际会议》。施普林格, 2021年, 第92–107页。[5] Mavin A. 需求语法简易方法（EARS） / A. Mavin 等 // 2009年第17届IEEE国际需求工程会议（RE）。– 美国佐治亚州亚特兰大：IEEE, 2009。– 第317-322页。[6] A. Veizaga、M. Alferez、D. Torre、M. Sabetzadeh和L. Briand, 《系统化构建功能需求受控自然语言》, 《实证软件工程》, 第26卷, 第4期, 第79页, 2021年。[7] Sascha Konrad和Betty H. C. Cheng。2005年。实时规格模式。载于第27届国际软件工程会议论文集（ICSE '05）。计算机协会，纽约，美国，372–381。https://doi.org/10.1145/1062455.1062526[8] Dwyer M. B.、Avrunin G. S.、Corbett J. C. 有限状态验证中性质规格说明的模式 //第21届国际软件工程会议论文集。– 1999。– 第411-420页。[9] Gustavo Carvalho、Diogo Falcão、Flávia Barros、Augusto Sampaio、Alexandre Mota、Leonardo Motta和Mark Blackburn。2013年。基于SCR规格说明从自然语言需求生成测试用例。载于第28届ACM应用计算年度研讨会论文集（SAC '13）。计算机协会，纽约，美国，1217–1222。https://doi.org/10.1145/2480362.2480591[10] Xi Zheng、Christine Julien、Rodion Podorozhny和Franck Cassez。2015年。BraceAssertion：信息物理系统的运行时验证。载于2015年IEEE第12届移动自组网与传感器系统国际会议论文集（MASS '15）。IEEE计算机学会，美国，298–306。https://doi.org/10.1109/MASS.2015.15[11] Filipovikj P.、Seceleanu C. 使用规格模式描述工业系统需求：从业者评估案例研究 //ENASE 2019：第14届软件工程新方法评估国际会议，希腊克里特岛伊拉克利翁，2019年5月4日至5日。– 2019。– 第92-103页。[12] Filipovikj P. 等。将基于模式的形式化需求规格说明集成到工业工具链中 //2016 IEEE第40届年度计算机软件与应用会议（COMPSAC）。– IEEE，2016。– 第2卷。– 第167-173页。[13] Stachtiari E. 等。通过构造正确性对系统需求与设计进行早期验证 //《系统与软件杂志》。– 2018。– 第145卷。– 第52-78页。[14] Marco Autili、Lars Grunske、Markus Lumpe、Patrizio Pelliccione和Antony Tang。2015年。使用结构化英语语法统一定性、实时和概率性质规格模式。《IEEE软件工程汇刊》41卷7期（2015年7月），620–638。https://doi.org/10.1109/TSE.2015.2398877[15] Claudio Menghi、Christos Tsigkanos、Thorsten Berger、Patrizio Pelliccione和Carlo Ghezzi。2018年。机器人任务的性质规格模式。载于第40届国际软件工程会议论文集：会议伴随论文集（ICSE '18）。计算机协会，纽约，美国，434–435。https://doi.org/10.1145/3183440.3195044[16] Becker J. S. 等。用于需求驱动的基于模型开发的可互操作工具链 //ERTS 2018。– 2018。[17] J. S. Becker, 《分析形式化需求一致性》, ECEASST, 第76卷, 2019年5月。[18] Becker, J.S.（2018）。使用Kind2模型检查器对基于模式的契约进行虚拟集成。载于：Howar, F., Barnat, J.（编）《工业关键系统形式化方法》。FMICS 2018。《计算机科学讲义》，第11119卷。施普林格，查姆。https://doi.org/10.1007/978-3-030-00244-2_9[19] Adeniyi O. A. 具有时序特征的受控自然语言 / O.A. Adeniyi。– 曼彻斯特大学，2018。[20] Getmanova A. N. 等。事件驱动时序逻辑需求的语义分类 //2022 IEEE第23届青年电子器件与材料专业人员国际会议（EDM）。– IEEE，2022。– 第 663-668 页。