

시스템 모델 (시퀀스 다이어그램) 문서 (System Model (Sequence Diagram) Document)

프로젝트명	과학과 기술 문서 이해를 돕는 웹 브라우저 확장 프로그램 개발
-------	------------------------------------

조	02 조
지도교수	정상근 교수님 (서명)
조원	201803851 양성욱 201802074 김재윤 202104514 박규수

Table of Contents

- [1. Introduction 1](#)
- [1.1. Objective 1](#)
- [2. Use Case Diagram 2](#)
- [3. Sequence Diagram 3](#)
- [3.1. AMSM_REQ_Monitoring_N001 \(SubscribeESEStatus\) 3](#)

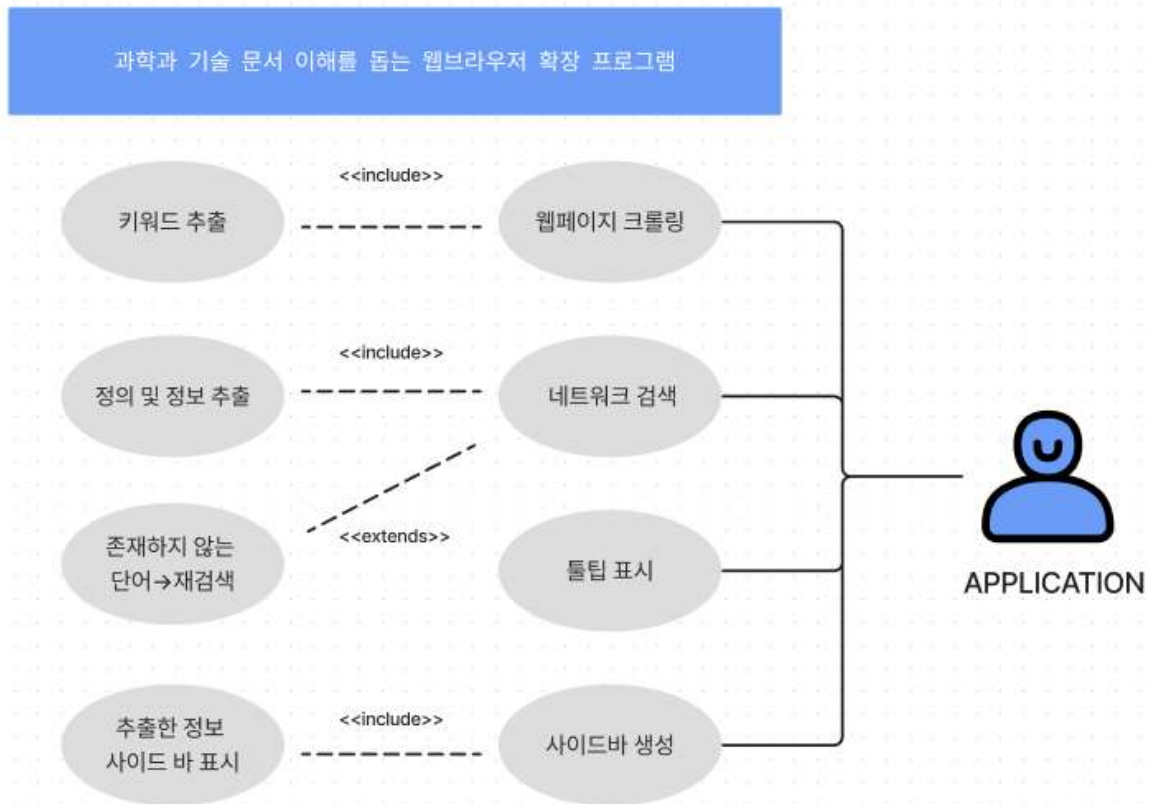
1. Introduction

1.1. Objective

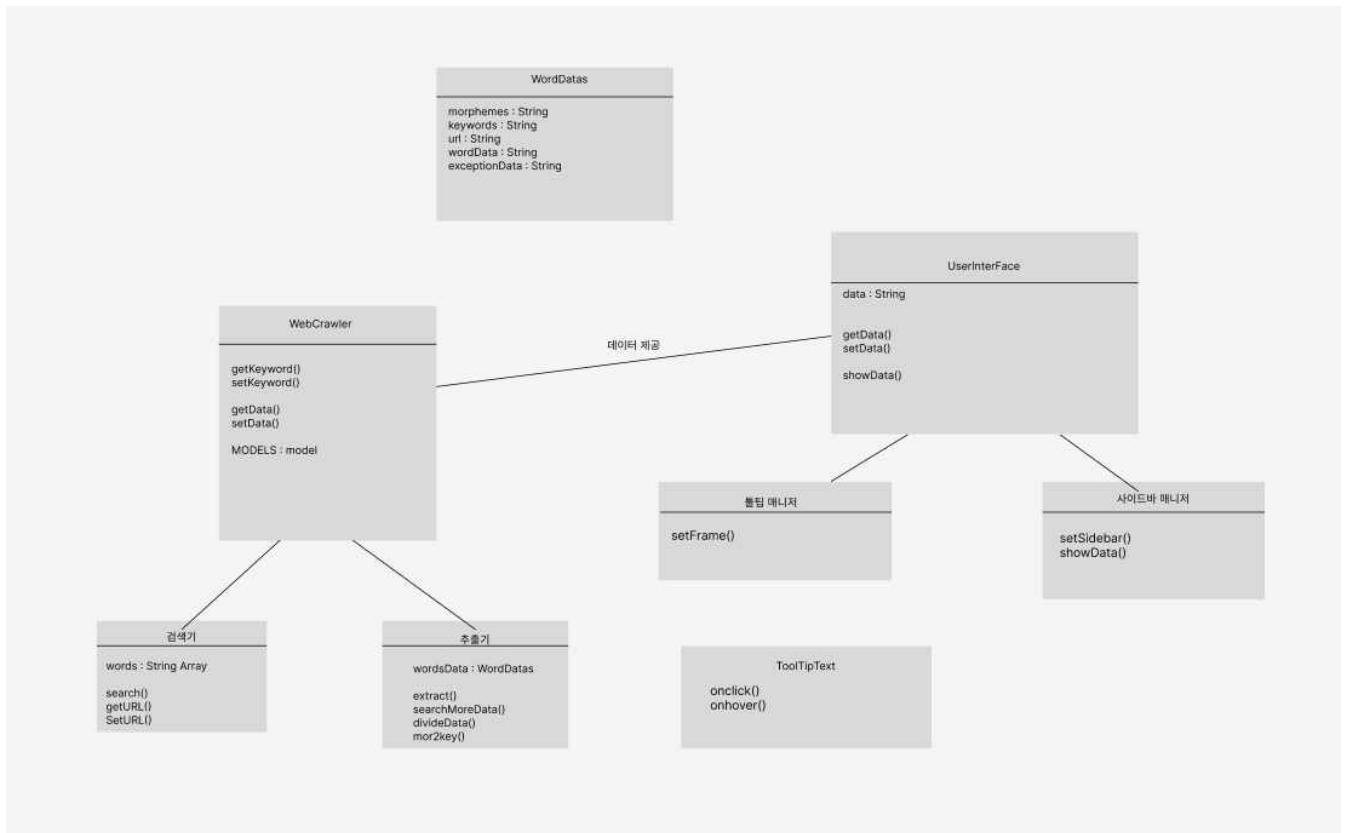
-이 문서는 과학과 기술 문서 이해를 돕는 웹 브라우저 확장 프로그램 개발에 대한 시스템 모델(시퀀스 다이어그램)에 대한 내용을 기술한다.

-이 문서에서는 요구사항 명세 단계에서 작성한 유스케이스 다이어그램과 클래스 다이어그램 기반으로 각 클래스간의 상세한 내부 동작흐름을 시퀀스 다이어그램으로 모델링한다.

2. Use Case Diagram



3. Class Diagram



4. Sequence Diagram

4.1. 웹 크롤링

이름	웹 크롤링
<div> <div>sd Web Page Crawling</div> <pre> sequenceDiagram actor User participant Extractor as 추출기 participant WordDatas as WordDatas : wordDatas User->>Extractor: extract() Extractor->>WordDatas: divideData() Extractor-->>User: if extract() fail sendError() </pre> </div>	
기능	사용자가 보고 있는 웹페이지의 본문 내용을 추출하여 텍스트데이터로 가져온다.
동작 순서	<ol style="list-style-type: none"> 1. 현재 사용자가 사용하고 있는 웹페이지의 본문 내용을 추출한다. 2. 추출한 내용을 분류하여, 데이터를 "wordDatas"에 저장한다. 3. 내용 추출에 실패 시, 사용자에게 실패하였다는 오류 메시지를 전달한다.

4.2. 키워드 추출

이름	키워드 추출
<div> <div>sd Keyword Extract</div> <pre> sequenceDiagram participant User participant WordData as WordData: wordData participant Extractor as 추출기 User->>WordData WordData->>Extractor: wordsData : wordData.getMorphemes() Extractor->>Extractor: mor2key() Extractor->>WordData: wordData.setKeywords() WordData-->>User: return otherSuggestion() </pre> </div>	
기능	텍스트 데이터에서 사용자에게 정의를 전달할 키워드를 추출한다.
동작 순서	<ol style="list-style-type: none"> 1. 저장된 데이터를 형태소로 나눈다. 2. 형태소로 나뉜 데이터에서 키워드를 추출하여 저장한다. 3. 키워드 추출에 실패 시, 사용자에게 다른 키워드를 추천, 제공하거나 사용자가 직접 키워드를 입력할 수 있도록 안내한다.

4.3. 네트워크 검색

이름	네트워크 검색
	<pre> sequenceDiagram participant S as 검색기 participant M as MODEL participant W as WordDatas:wordDatas M->>W: Keyword:getKeyword() S->>M: searchSite() M->>M: researchSite() M->>W: opt-can't find site W->>M: Exception: 재검색 M->>W: Exception: 재검색 M->>W: setURL() </pre>
기능	추출된 키워드를 바탕으로 적절한 정보를 가진 사이트의 URL을 받는다.
동작 순서	<ol style="list-style-type: none"> 1.wordDatas에서 keyword를 받아온다 2.받아온 keyword를 모델에 넣고 search()를 진행한다. 3.적당한 사이트가 없을경우 research()과정을 진행한다. 4.반복하여 없을경우 exception:재검색 과정을 실시한다. 5.적절한 사이트가 있을경우 URL을 wordDatas에 저장한다.

4.4. 정의 및 정보 추출

이름	정의 및 정보 추출
	<div data-bbox="161 394 277 412" style="background-color: #f08080; padding: 2px; margin-bottom: 10px;">정의 및 정보 추출</div> <pre> sequenceDiagram participant S as 검색기 participant E as 추출기 participant W as WordDatas:wordDatas S->>E: extract() E->>W: URL:getURL() W-->>E: searchMoreData() E->>W: setData() </pre>

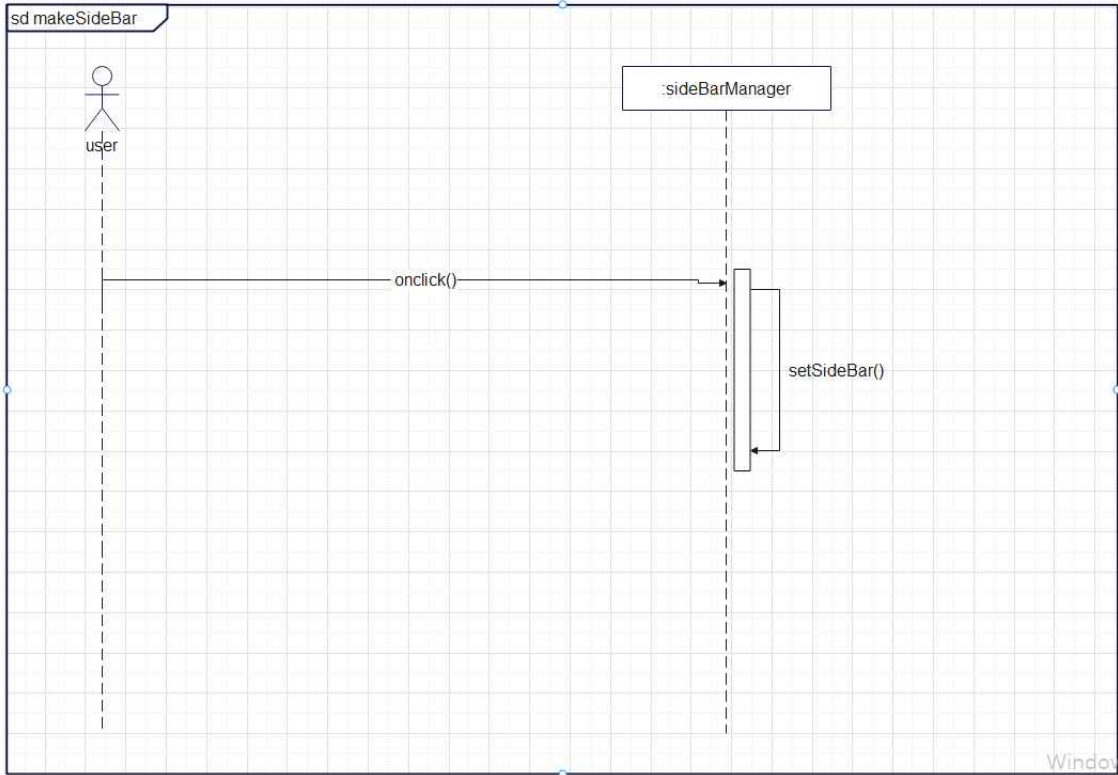
4.5. 존재하지 않는 단어(재검색)

이름	존재하지 않는 단어(재검색)
<div>존재하지 않는 단어(재검색)<Exception case></div> <pre> graph TD MODEL[MODEL] --> wordAnalyze(wordAnalyze(Keywords)) wordAnalyze --> opt[opt] subgraph opt_block [opt] direction TB S1(getNounType(Keywords) if word != noun) S2(getAcronym(keywords) if word == Acronym) S3(getTranslateKorToEng(keywords) if word == korean) S4(getSimilar(keywords) else) end opt_block --> MODEL </pre>	
기능	keywords를 LLM으로 처리하여 새로운 형태의 Keywords로 받아온다.
동작 순서	<ol style="list-style-type: none"> 1.LLM에 해당 keywords를 넣어 분석한다. 2.해당 조건에 걸맞는 형태중 하나를 opt처리하여 받는다. <ol style="list-style-type: none"> 2-1)만약 해당 단어가 명사형이 아니면 명사타입으로 바꿔 반환한다. 2-2).만약 해당 단어가 줄임말형이라면 풀어서 반환한다. 2-3)만약 해당 단어가 한국어라면 영어로 번역하여 반환한다. 2-4)만약 해당 단어와 유사한 단어가 있다면 반환한다.

4.6. 툴팁 표시

이름	툴팁 표시
	<pre> sequenceDiagram actor User participant TTM as :ToolTipManager participant WD as WordDatas:wordDatas participant TT as ToolTipTexts : tooltipText User->>TT: onhover() activate TT TTM->>WD: data = getData() activate WD WD-->>TTM: deactivate WD TTM->>TT: setFrame() activate TT TT-->>TTM: deactivate TT deactivate TT </pre>
기능	<p>정의를 얻어온 단어에 대해 배경색 표시를 해주고, 마우스를 hover시켰을 때, 그 정의를 상단 말풍선에 표시하도록 설정한다</p>
동작 순서	<ol style="list-style-type: none"> 1. 네트워크 검색 이후 정의를 얻은 단어들(wordDatas)을 가져온다 2. 정보를 모두 가져온 후 해당 단어들에 배경색 표시를 한다. 3. 배경색 표시된 단어들에 마우스를 hover시키면 단어의 정의가 상단 말풍선에 표시된다.

4.7. 사이드바 생성

이름	사이드바 생성
	 <pre> sequenceDiagram participant user participant sidebarManager as :sidebarManager user->>sidebarManager: onclick() activate sidebarManager sidebarManager->>sidebarManager: setSideBar() deactivate sidebarManager </pre> <p>The diagram is a UML sequence diagram titled 'sd makeSideBar'. It features two lifelines: 'user' (represented by a stick figure) and ':sidebarManager' (represented by a rectangle). The process begins with a message 'onclick()' sent from the user to the sidebarManager. This triggers an activation bar on the sidebarManager lifeline. Within this activation, a self-call message 'setSideBar()' is shown. The diagram is set against a grid background with a 'Windows' logo in the bottom right corner.</p>
기능	사용자가 툴팁표시된 단어를 클릭시 사이드바가 생성된다.
동작 순서	<ol style="list-style-type: none"> 1. 배경색이 표시된 단어를 클릭할 시, 사이드 바를 생성한다. 2. 사이드 바가 생성된 이후, 내부 정보를 준비한다.(setSideBar())

4.8. 추출한 정보 사이드 바 표시

이름	추출한 정보 사이드 바 표시
	<pre>sequenceDiagram participant S as :sideBarManager participant W as WordDatas: wordDatas S->>W: data : getData() activate W W-->>S: deactivate W S->>S: setSideBar() activate S S->>S: deactivate S</pre> <p>The diagram is a UML sequence diagram titled 'sd setSideBar'. It features two lifelines: ':sideBarManager' and 'WordDatas: wordDatas'. The process begins with a message 'data : getData()' sent from ':sideBarManager' to 'WordDatas: wordDatas'. This message is represented by a solid arrow. The 'WordDatas: wordDatas' lifeline is active during this period, indicated by a solid vertical bar. After the message is received, there is a return message (represented by a dashed arrow) from 'WordDatas: wordDatas' back to ':sideBarManager'. Following this, the ':sideBarManager' lifeline performs a self-call 'setSideBar()', shown as a solid arrow looping back to its own activation bar. The diagram is set against a light gray grid background.</p>
기능	클릭한 단어에 대해 가져온 정보를 사이드 바에 표시한다.
동작 순서	<ol style="list-style-type: none">1. 클릭한 단어에 대한 정보를 가져온다.2. 가져온 정보를 사이드바 내부에 출력한다.