<6장> 서면 형식으로 프로젝트 제시하기(나타내기)

# 목표 : 서면 형식으로 프로젝트를 효과적으로 나타내기 위한 기술들을 소개한다.

# 배우는 것들

- 전문적인 보고서를 구성하고 작성하는 법을 배운다.

- 분명하고 간결한 개요를 작성한다.

- 데이터와 결과를 명확하게 제시하는 방법을 배운다.

- 표절을 피하고 자료를 참조하는 방법을 배운다.

- 소프트웨어를 기록하고, 프로그램은 논평하고 사용자 가이드를 작성한다.

[6.1] 소개

당신의 아이디어와 결과의 배포는 연구 프로젝트 과정의 중요한 부분으로 확인되었다. 상당한

수준의 소프트웨어를 개발하지 않았다면, 보고서가 해당 프로젝트의 유일한 증거가 될 것이다.

보고서가 프로젝트의 전체적인 모습을 보여주기 때문에 보고서 작성을 소홀히 한다면, 프로젝트

자체를 망치게 될 수 있다는 것을 기억해야 한다. 자신이 수행한 결과를 다른 사람에게 제대로

보여주지 못한다면, 그동안 노력한 엄청난 양의 연구 개발과 그것들의 가치는 의미가 없게 된다.

그렇다고 해서 반대로 좋은 보고서를 작성한다고 해서 프로젝트 자체가 좋아진다거나 하는 일은

없다. 물론 좋은 보고서를 작성함으로써 어느정도는 프로젝트를 개선시킬 수 있지만, 항상

보고서가 프로젝트의 반영이고 몇 가지 말로 허술한 조사, 개발, 분석 및 방법을 위장할 수

없다는 것을 명심해야 한다.

이 챕터는 프로젝트에 대한 서면 자료 제시에 초점을 맞춘다. 보고서 구성, 개요 작성, 자료

참고, 그리고 데이터 표현 등을 다룬다. 또한 소프트웨어 기록, 프로그램 논평, 그리고 사용자

가이드 작성과 같은 주제들은 다룬다. 발표 자료와 viva를 통해 구두 형식을 어떻게 발표하는지가

다음 장의 주제이다.

[6.2] 보고서 구성 및 작성

[6.2.1] 고려 사항

프로젝트 보고서 작업을 시작하기 전에 염두에 두어야 할 두가지 주요 고려 사항이 있다.

- 독자가 누구인가? 그들이 이미 알고 있는 것은 무엇인가? 그들이 알고 싶어하는 것, 배우고

싶어 하는 것은 무엇인가? 그들이 보고서를 통해 얻고 싶은 것은 무엇인가? 그들에게 어떤

영향을 주고 싶은가?

- 보고서 양은 어느 정도 되어야 하는가? 속해 있는 학교나 기관이 보고서 길이 제한을

두었는가? 또한 그 길이에 딱 맞춰야 하는가? 당신이 수행할 작업과 얻을 수 있는 결과물에

기초하여 합리적인 보고서 길이(양)는 어느 정도인가?

이러한 고려사항들은 보고서에 포함시킬 것들이나 마지막 최종 보고서에서 누락시킬 수 있는

것들에 영향을 줄 것이다. 기입할 자료들이 독자들을 성가시게 할 수 있고 또 그들이 불필요하게

느껴질 수 있기 때문에 그 자료들을 단순히 그 자체를 위해 포함시켜서는 안된다. 또한 당신이

중요하다고 생각하는 자료들은 절대 누락시켜선 안된다. 넣어야 할 것들은 반드시 포함시키고,

굳이 넣지 않아도 될 내용은 뺌으로써 자료들의 균형을 이루도록 해야 한다. 당신이 말하려 하는

것을 이해하고, 독자들이 이미 갖고 있는 지식에 대해 인식하여, 이러한 상황을 고려하여

자료들을 적절하게 기입하고 배치해야 한다.

[6.2.2] 보고서 작성법

사람들이 보고서를 작성할 때 자주 사용하는 두가지 주요 접근법(방법)이 있다. 하향식(top-down)

방식과 진화적 출시 모형(evolutionary delivery)이다. 이 두가지 접근법은 상호 배타적이지 않고,

아마 프로젝트를 진행하면서 두 접근법을 어느정도 이용하게 될 것이다.

하향식 접근법은 보고서 몇 장의 챕터로 구성되어 있는지, 각 챕터들은 어떤 내용을 담고

있는지, 그리고 각 챕터들은 하위 섹션들로 어떻게 분해되는지 등의 구조를 파악하는데 사용된다.

이러한 접근법을 사용함으로써 각 챕터의 세부적인 섹션들을 파악할 수 있다. 부제목을

파악함으로써 그것들을 적절한 시점에 마무리하고 그것들로부터 결과를 얻을 수 있다. 6.1절은

특정 챕터에 대한 분해 예시를 보여주고 있다. 분해된 각 덩어리들의 내용을 이해함으로써,

각 섹션(덩어리)들이 한번에 하나씩 다뤄지기 때문에 글을 쓰는 사람으로 하여금 글 작성을 더

용이하게 해주고 덜 망설이게 해준다. 전반적인 챕터 구조를 파악함으로써 전반적인 목표에

집중하게 해주어 접선을 따라 시작하지 않고 요점에 맞지 않는 관계 없는 아이디어들은 논할 수

있게 한다. 챕터 분해는 또한 작성해야 하는 보고서 양에 대해 더 잘 이해할 수 있게 함으로써

시간 관리 측면으로도 도움을 준다. 이는 각 섹션의 복잡성에 대한 이해로부터 기인하고, 이는

이러한 섹션들을 완성하는데 얼마나 걸리지는 지에 대한 예측을 제공해주기도 한다.

보고서의 여러 섹션들과 그에 대한 하위 섹션들에 대한 이해와 파악은 프로젝트 초기에

이루어져야 한다. 하지만, 흔히 있는 일이지만, 보고서에 포함시켜야 하는 것을 완전하게

이해하고 모든 챕터의 상세한 내용들을 파악할 수 있고 나서야 비로소 프로젝트를 완성할 수

있다. 어찌됐든, 보고서 분해 구조는 당신의 사고와 아이디어들은 구성하고 그것들은 보고서

내에서 서로 연관지을 수 있는 유용한 방법임을 알게 될 것이다.

보고서 작성에 사용되는 또 다른 방법은 진화적 출시 모형이다. 많은 사람들이 이 방법을

사용하지만, 그들이 사용하고 있는 것을 의식하지 못하고 있다. 이 방식에서 당신은 우선

보고서 일부를 작성하고 이 부분들은 프로젝트를 진행하면서 계속해서 재작성한다. 따라서

각 부분들은 새로운 아이디어가 생기고 그것들에 대한 이해도가 증가함에 따라 계속해서

보완되고 개선되게 된다. 따라서, 프로젝트를 마쳐도 가만히 있지 못하고, 보고서를 일회성으로

쓸 수 없게 된다. 프로젝트 생명 주기 동안 계속해서 재작성하고 보완하고 개선하는 작업을

반복해야 한다.

앞서 소개한 두가지 방법들은 혼합되어 사용될 수 있어서 아마 프로젝트 초기에도, 보고서의

특정 챕터들을 파악할 수 있을 것이다. 그런 뒤 이 특정 섹션들을 작성할 수 있게 되지만,

프로젝트 진행에 따라 그것들이 진화하고 변화했음을 알게 될 것이다. 또한 이해도가 증가하고

아이디어가 변화하고 발전하며, 그에 따른 결과들은 얻게 됨으로써 보고서 분해 구조가 시간이

지남에 따라 그 자체로 진화했음을 알게 될 것이다.

[6.2.3] 보고서 작성 순서

프로젝트 끝에 마지막 글 작성을 마쳤든 프로젝트를 진행하면서 보고서를 개선시키고 있든,

반드시 지켜야 할 글 작성 순서가 존재한다. 이 순서는 다음과 같이 나눠진다.

\* 구조를 파악해라. 보고서 분해 구조를 사용함으로써 이는 폰트 크기와 유형, 페이지 크기,

순서 매기기 등의 관점에서 보고서 내용과 보고서 형식 모두와 연관이 있다.

특정 내용 구조가 이 단계에서 명확하진 않지만, 당신은 가능한 한 상세한 챕터 분해를

만들어내야 한다.

\* 발표와 관련된 스타일을 파악해라. 이 단계에선 보고서의 발표와 관련된 측면들에 대한 표준,

즉 레이아웃을 지정해야 한다. 이는 이후에 분해한 챕터들과 섹션들의 순서를 정하거나

순서가 잘못되어 시정이 필요한 경우에 도움을 줄 수 있다. 가이드 라인이 있다면 따라야 하고

만약 가이드 라인을 가지고 있지 않다면, 보고서 레이아웃을 위한 다음의 고려사항들을

참고해야한다.

-> 너무 넓고(텅빈) 오픈된 공간이나 너무 비좁아 알아보기 힘든 레이아웃을 피해야 한다.

숫자나 그림, 혹은 테이블이 텍스트의 공간이 텍스트를 방해하지 않도록 해라.

-> 11 혹은 12 폰트 크기를 사용해라. Times 나 Genova와 같이 읽기 쉬운 것들을 사용해라.

-> 결합하기에 적합한 여백의 단일 맞춤 열을 사용해라. 하지만, 두 개의 복잡 열 사용이 텍스트

줄을 짦게 하기 때문에 읽기에 더 유용하다는 말이 있기 때문에 이 부분에 대해서는 당신의 기관

가이드라인을 참고하거나 지도 교수의 조언을 따라라.

-> 제목 아랫글 영역의 중앙에 페이지 번호를 넣도록 해라.

줄 간격, 섹션 번호 매기기, 문단 스타일 등에 있어서 본인만의 스타일을 정의하고 싶어할지도

모른다.

오늘날 칼라 프린트의 확산으로, 텍스트를 강조하기 위해 어떤 텍스트 칼라를 사용할지

결정하고 싶어할지도 모른다. 빨간색 챕터 제목, 파란색 섹션 헤더 등과 같이 복잡한 스타일을

지정하지 않도록 해야한다. 이러한 형식의 제시는 번잡해 보일 수 있기 때문에 보고서 대부분의

텍스트를 검정색으로 하는 것이 가장 좋은 방법이다. 하지만 색깔은 테이블이나 특정 부분들을

강조하는 데 효과적으로 사용되곤 한다. 예를 들어, 통계 데이터를 나타내는 테이블의 각기 다른

부분들을 구별하기 위해 중요 결과들을 강조하는 색을 사용하고 싶어할 수도 있다.

\* 서론 초안을 작성해라. 서론(도입부)는 독자로 하여금 보고서 내용에 대한 아이디어 제공해주기

때문에 당신의 생각을 분명히 하는 데 도움을 준다. 하지만, 이 단계에서, 당신이 프로젝트를

완료하는 시간에 따라 강조점이 변화하고 당신의 아이디어는 진화해야 할 의무가 있기 때문에

도입부는 초안이 될 수 밖에 없다. 따라서 도입부는 문헌 검토를 포함하거나 이들로 구성되어야

할지도 모른다는 것을 기억해라. 또한 주제에 대한 기초가 성립될 수 있도록 초기에 다뤄져야

한다.

- 본론. 보고서의 본론은 다음 작업해야 할 부분으로서 사용된 방법들, 수행된 분석 등과 같은

챕터들이 포함될 것이다. 분명, 시작한 프로젝트에 따라 본론의 내용이 달라질 것이다. 프로젝트

진행 상황에 따라 본론의 일부를 작성하고 각 챕터와 섹션들을 필히 순서대로 작성하지 않음을

알게 될 것이다. 보고서에 명시한 것과는 다른 순서로 각기 다른 시간에 프로젝트의 부분들이

완성될 것이다.

\* 결론과 추천. 결론과 추천이 마지막에 완성하게 되는 것들 중 하나일 것이다. 프로젝트가

완성되면 당신은 성취한 것과 마지막 생각이나 추천을 나타낼 수 있는 것에 대해 완전히

이해하게 된다.

\* 서론을 완성해라. 진화적 작성 방법의 일부로서, 당신은 프로젝트 보고서를 모두 완료한 후

서론에 대해 다시 작성할 필요가 있음을 알게 될 것이다. 마지막 결과를 암시하는 텍스트를

포함시키거나 더 세부적으로 집중해왔던 주제에 대한 배경을 더 소개하고 싶어할지도 모른다.

\* 요약(abstract)을 작성해라. 보고서에 포함된 것들에 대해 알게 되기 전까진 요약을 확실하게

작성할 수 없을 것이다. 요약을 효과적으로 작성하는 방법은 이후에 이 챕터에서 다뤄질 것이다.

\* 참조와 부록. 프로젝트를 진행하면서 참조와 부록들을 수집하고 있을지라도, 보고서 나머지가

완성되기 전까지는 참조와 부록의 제시를 완성해선 안된다. 참조는 추가되거나 삭제될 수 있고,

부록에서 자료들을 포함시키거나 제거할 수도 있다

\* 내용 목록과 인덱스를 배열해라. 색인 및 목록을 끝까지 남겨라. 그래야 보고서의 정확한

내용과 모든 페이지 번호를 알 수 있다.

\* 교정하고 확인하고 수정해라. 보고서가 완성된 뒤 다시 교정하는 작업을 매우 중요하고

필수이다. 종종, 보고서를 오랜 시간동안 너무 가까이 해서 바로 보고서를 읽는 것은 오류와

누락을 잘 못 찾게될 수 있음을 의미한다. 쓰여져 있든 안 쓰여져 있는 본인은 의도를 알기

때문에 정확히 파악할 수 없다. 이러한 의미에서 교정하기 전에 잠시 보고서와 떨어져 있는

것도 좋은 방법이 될 수 있다. 이렇게 하면 프로젝트를 교정하고 수정하고 발생하는 어떤

요점들에 변화를 주는 등의 작업을 프로젝트 제출 전의 몇 일동안 진행할 수 있음을

기억해둬라.

[6.2.4] 구조

보고서는 다음과 같은 섹션들로 구성되어야 한다.

\* 제목 페이지. 제공되는 가이드 라인을 따라라. 최소한 제목, 저자, 날짜와 학위 수여 정보는

포함시켜라.

\* 요약

\* 프로젝트를 위해 도움을 준 사람들에 대한 감사

\* 내용 목록, 목차

\* 숫자, 그림, 테이블의 목록. 이는 필수 사항이 아니고 이 자료들이 보고서에 가치를 더하고

독자들에게 유용하다고 생각될 때만 이것들을 포함시켜라.

\* 보고서 그 자체

-> 소개 / 문헌 검토. 보고서의 첫 장은 소개로 구성되어야 한다. 자주, 소개하는 장이 문헌

검토를 제시하는 역할을 하곤 한다. 대신에, 소개 및 도입부는 프로젝트가 보고서의 대략적인

개요와 같은 역할을 하고 문헌 검토는 그 자체 별개의 장으로 나중에 제시된다. 소개부는

프로젝트 보고서의 장면을 설정하고, 프로젝트의 목적과 목표들을 포함해야 한다.

-> 본론

-> 결론/추천

\* 적절한 형시의 참조. 참고하는 자료는 이후 더 자세하게 논의된다.

\* Appendix A, Appendix B, Appendix C등과 같이 처리된 부록. 이는 프로그램 목록, 테스트 결과,

그리고 초기 제안서, 프로젝트 계획, 회의 보고서 등과 같은 프로젝트 세부 사항들을 포함한다.

\* 필요 시 용어 해설 제공

\* 필요하다면 색인을 추가하지만, 가능한 제외해라.

[6.2.5] 스타일

보고서를 나타내기 위해 택한 작성 스타일은 세가지 관점으로 논의될 수 있다. 첫번째는

보고서의 실제 표현 스타일이다. 예를 들어, 레이아웃이나, 폰트 등이 있다. 이러한 종류의

스타일은 앞서 논의되었었다. 두번째는 보고서 내에서 사용하는 문법적 스타일이다. 종종

저급한 문법 사용으로 좋은 보고서를 망치곤 한다. 과도한 단어와 전문 용어가 포함된 길고

복잡한 문장 안에 아이디어와 결과가 드러나 있지 않고 숨겨져 있기 때문에 저자의 뜻하는

바나 의도가 불분명해진다. 세번째 관점은 전반적인 내용 구조로, 이는 이후 논의된다.

좋은 작성 스타일은 연습으로부터 비롯된다. 쓰면 쓸수록 더 분명하게 좋은 스타일이 나타난다.

읽기 또한 좋은 실행 요소를 배우고 논쟁들을 표현하고 논의하는 흥미로운 방법들을 식별할 수

있게 해줌으로써 글쓰기 능력을 향상시켜 준다. 이렇게 말했지만, 전문 보고서 작성 스타일을

향상시키기 위해 따라할 수 있는 몇 가지 간단한 규칙들이 있다. 나, 너, 우리, 나의 과 같은

개인 대명사 사용을 피해야 한다. 하지만 단지 이것을 피하기 위해 정교하고 복잡한 문장을

만들지 마라. 지도 교수가 이에 관한 조언을 해줄 것이고 그것이 개인적인 방법을 사용하는

필요한 프로젝트의 본질일 수 있다. 문장을 간결하고 요점 위주로 작성해라. 같은 문장 내에

여러 가지 포인트를 포함시키지 마라. 약어, 전문 용어, 그리고 속어 사용을 피해라. 복잡한

단어보다는 단순한 단어를 사용해라. 복잡한 것들을 독자들을 괴롭히고 문장의 본 뜻을 흐리게

한다. 이를 어설프게 사용하면 그 주제에 대한 자신의 이해 부족을 감추기 위해 사용된 것으로

보일 수 있고 이는 또 배운 독자들에 의해 드러나게 되기도 한다.

보고서는 이미 완성한 프로젝트의 결과를 나타내기 때문에 보통 과거 시제를 사용하는 것이

일반적이다. 이렇게 말하면서, Day는 현재 시제를 타인의 작업을 언급할 때 사용해야 한다고

제안한다. 농담과 사설을 피해라. Isn’t와 같은 축약 표현을 피하고 대신에 is not과 같은 표현을

사용해라. 어포스트로피의 사용법을 제대로 숙지해라. 마지막으로 스펠링 체크를 해라. 이런

부분에서 미흡한 점을 드러내면 보고서의 겉 이미지가 안좋아 진다.

기본 문법에서 벗어나 보고서 작성 시 고려해야할 세번째 유형은 전반적인 내용 구조이다.

-> 시작 : 소개와 문헌 검토가 장을 구성한다.

-> 중간 : 프로젝트 주요 구성 요소가 포함된 보고서의 주 부분

-> 끝 : 결론, 요약, 권고 그리고 이후 작업

이러한 유형의 구조는 보고서의 각 장에서 명백하게 나타난다. 소개(가능하면 개요)와 본론과

끝(가능하면 요약이나 결론)이 반드시 포함되어야 한다.

[6.2.6] 팁

보고서 작성에서 이 섹션은 당신에게 도움이 될만한 몇가지 팁을 제공한다. Bell은 작성 능력을

개선시키고 스스로 훈련하는 데 도움을 줄 몇가지 포인트들을 말한다.

\* 마감기한을 설정해라. 당신의 보고서는 완성하는 데 오랜 시간이 걸린다. 마감 기한을

설정하지 않고 그것들에 매달리기만 한다면, 분명 제 시간에 끝내지 못할 것이다. 보고서 분해

구조는 시간을 계획하는 데 도움을 줄 것이다.

\* 규칙적으로 작성해라. 하루 중에 가장 작성하기 좋은 시간과 장소를 선택하여 작업해라. 다시

말해서, 마음이 괜찮을 때 작성학도 규칙적인 장소를 정해라. 사람들은 산만하거나 극도로 지친

상태에서 글을 쓸 수 없다.

\* 작업 리듬을 만들어라. 일단 보고서 작성을 시작하면, 계속해서 진행해라. 텍스트를 쓰고

있는 중이라면, 다른 것을 확인하기 위해 멈추지 말고 휴식을 취할 시간까지 계속 작성해라.

\* 준비가 된 섹션을 작성해라. 이는 보고서 작성이 기존 텍스트의 짜집기에 불과하고 소개와

결론을 도출하는 것에 지나지 않을 프로젝트 막바지까지 시간을 절약해 줄 것이다.

\* 다시 재개하기 쉬운 부분에서 멈춰라. 쉬고 난 뒤에 다시 작업을 재개하기 쉽지 않고 또 어느

정도의 시간이 걸린다. 예를 들어 한 섹션 전체를 완성했을 때 잠시 쉬어라. 작성을 멈추고

하루 혹은 한 주간 쉬고 난 뒤 다시 돌아와 글을 이어가려 할 때, 본인이 해당 부분에서

쓰려고 의도했던 바를 기억하기란 쉽지 않을 수 있다. 그로 인해 글의 흐름이 꺠질 수도 있기

때문에 이런 상황에 대비에 항상 글 작성을 멈추기 전에는 해당 부분에 대한 필요한 메모를 해서

이후 글을 이어 작성할 때 도움이 될 수 있는 환경을 마련해놔라.

따라하면 좋은 또 다른 팁은 글을 쓰기 전에 함께 필요한 모든 자료들을 수집해놓는 것이다.

자료를 검색하거나 서적을 참고하기 위해 도서관을 방문하려고 글의 작성 흐름을 깨는 것은

도움이 되지 않는다.

컴퓨터 관련 학생들에겐 더 말할 것도 없이 워드 프로세서를 이용해 글을 작성하는 것이 가장

좋은 방법일 것이다. 이러한 패키지들은 다른 작성 방식에 비해 훨씬 효율적일 것이다. 대부분의

워드 프로세서들은 사전들과 같은 여러 내장 기능들을 제공한다. 게다가,방정식 편집기를 제공해

깔끔한 방정식을 만들 수 있게 도와준다. 대안으로, 방정식 편집기는 보고서에 붙여넣기 전에

방정식을 만드는 데 사용될 수도 있다. 다음의 예시는 보고서에 붙여넣기된 방정식이다. 이

방정식이 어떻게 옆에 명시된 참조 번호를 부여받았는 지 주목해라. 이 숫자는 항상 보고서에

통합되는 각 방정식을 고유하게 식별하기 위해 포함해야 한다.

내장된 스펠링 체크 기능을 사용할 때 주의해야 한다. 대부분이 미국 영어 사전에 기반한

기능이어서 미국 영어로 변환시켜 줄 것이다. 스펠링 체커는 인용한 문구 안의 내용에서

발생한 오류도 잡아낼 수 있다.

문법에 관련해서도 자동적으로 오류를 발견하고 수정해준다. 그러나 주의해야 할 것이

문제없어 보이고 잘 만든 문장같은데도 문법 오류를 나타내고 수정을 요하며 자동적으로

문법 변환을 하는 경우가 있는데, 이러한 경우엔 검사 기능이 약해보이므로 기능은 무용지물이

된다. 따라서 자가 체크하는 것이 나을 수도 있다.

[6.3] 요약 작성하기

Blaxer는 abstract 의 기능을 연구 프로젝트의 성격, 맥락, 수행 과정, 그리고 주요 결과가

무엇인지를 간략하게 요약하는 것이라 정의한다. 요약은 독자에게 개요를 제공해주고 보고서

전체를 읽을지 안 읽을 지 결정하게 하는 역할을 한다. 따라서 요약은 간결하고 분명하며,

흥미로워야 한다.

많은 요약 부문이 목차처럼 구성되지만, 이는 독자들에게 거의 가치가 없는 것으로, 독자들은

단지 글의 종류를 위해 보고서의 내용 목록을 본다. 당신이 무엇을 성취했는지, 그리고 보고서의

내용이 무엇인지 알 때, 보고서 요약은 마지막 글들 중 하나가 되어야한다. 독자가 보고서를

검색하여 보고서를 찾거나 인용한 작성자와 친숙해지기를 원하는 것이 아니므로, 요약 참고

문헌을 사용하지 마라. 또한, 전문 용어와 약자의 사용을 피해라. 좋은 요약 부문을 작성하는

것은 연습으로부터 나온다. 좋고 나쁜 요약 표현에 대한 느낌을 얻으려면 다른 이들이 똑 같은

기사에 대해 요약 구성하는 방법에 유의해라. 다음의 예시는 소프트웨어 개발 비용을 예측하기

위한 인공 신경망 접근법에 기초한 동일한 기사들에 대한 요약본들이다.

첫번째 요약은 목차와는 다르게 표현된 반면, 두번째는 기사에 대한 장면을 설정하고 그 내용과

기사가 만든 기여도에 대해 표현한다. 첫번째 요약은 글의 섹션에 대한 세부 사항으로 제시되며,

약어와 독자에게는 생소할 수 있는 참고 자료를 포함한다. 요약을 작성할 때는 여기서의 두번째

요약본과 같은 구조와 스타일을 따라할 수 있도록 해야 한다.

[6.4] 자료 표현

[6.4.1] 소개

거의 모든 프로젝트에서 당신은 설문이나 조사, 소프트웨어 테스트 결과, 알고리즘 속도 시험

등으로부터 얻은 데이터들을 한 가지 형식 또는 또 다른 형식으로 나타내야 한다. 숫자 결과를

텍스트 형식으로 표현하는 것은 종종 수집된 정보에 대한 다소 건조한 해석을 제공할 수 있지만,

그래프나 차트 형식의 그림은 일어나고 있는 일에 대한 훨씬 더 만족스럽고 전체적인 아이디어를

제공한다. 다이어그램은 설명하기 어려운 복잡한 자료를 단순화시켜 줄 수 있다.

그림 자료가 수천 개의 단어와 같은 가치를 보일 수 있을지라도 택한 그림이 옳은 자료임을

확실히 해야하고 그것들의 진정한 의미를 숨길 수 있는 방법으로 결과를 제시해서는 안된다.

Mark에 의하면, 세 가지의 거짓말이 존재한다고 한다. 거짓말, 빌어먹을 거짓말, 그리고 통계이다.

당신은 통계적 결과를 당신이 원하는 방향으로 말하게 할 수 있다. 보고서를 작성할 때는

객관적이어야 하며, 결과를 명확하고 정직하게 제시해야 한다는 점을 기억해야 한다. 이 섹션에선

차트와 표를 이용해 정보를 제시하고, 가장 많이 사용되는 차트의 다양한 예를 제시하며, 차트가

잘못 사용된 일부 사례들을 보여준다.

[6.4.2] 차트와 그래프 표현

보고서에 포함하는 모든 형상과 표들은 숫자와 짧은 묘사와 함께 유일하고 확실하게

라벨링되어야 한다. 가장 일반적인 접근법은 현재의 장 번호 앞에 붙은 연속된 숫자들을

사용하여 각 그림과 표에 라벨링을 하는 것이다. ‘Figure 6.1 A chapter breakdown structure’ 을

예로 들 수 있다. 보고서 내에서 서로 다른 항목들을 나타내는 테이블과 이외 형상들은

Table 6.1’, ‘Figure 6.1’ 과 같이 동일한 번호로 표현하는 것이 허락됨을 알고 있어야 한다. 라벨을

붙이는 방식을 장마다 다르게 구성하지 말고 모두 동일하게 일관성을 유지하는 것이 좋다.

보고서에 숫자와 표를 사용할 때 그것들은 보고서에 어떤 가치를 더해줄 수 있기 때문에 반드시

포함시켜야 한다. 그것들은 단순히 멋져 보여서 포함되는 것이 아니어야 한다. 숫자와 표는

보고서 내에서 나타내고 있는 정보를 지원하고 명확하게 해줘야 하고 가능한 한 원래의 기준점에

가깝게 포함되어야 하지만, 이전에는 포함되지 않아야 한다. 예를 들어 Table 6.1을 보자. 이 표는

1998년에 컴퓨터 과정을 수료한 100명의 최종 학위 분류를 보여준다.

Table 6.1는 이러한 데이터를 텍스트만 사용하여 달성할 수 있는 것보다 훨씬 더 명확한

방법으로 제시한다.

Table 6.1이 위에 있는 텍스트 형식만으로 보여준 자료보다 따르기 훨씬 쉬울 수 있지만,

그렇다고 해서 그것이 반드시 데이터를 표현하는 최선의 방법이라는 것은 아니다. Figure 6.2가

이러한 결과를 해석하는 더 분명한 방법이고 학위 등급의 분포, 확산, 또는 패턴을 보다 전체적인

관점에서 보여준다. 데이터의 분포는 데이터가 최소한 순서형 구조를 보일 때만 관련이 있음을

알아야 한다. 다시 말해서, 데이터가 배열된 범주는 어떠한 범주의 증가된 크기 및 규모를

보여준다. 순서와 관련없는 클래스를 나타내는 데이터에는 분포가 없으며 이러한 차트의 열

순서는 중요하지 않다. 이 경우에서 차트는 각 범주 내에서 식별된 항목의 수 사이의 차이만을

강조할 수 있다.

Figure 6.2는 수직 막대 차트 혹은 컬럼(열) 차트이다. 이러한 차트들은 수평적으로도 표현할 수

있지만, 일반적으로 수직적 표현이 선호된다. 막대 차트는 범주 내 데이터를 나타내는 데

사용되고 Likert-type 척도 사용하는 질문의 결과를 표현하는 데 유용하다. 이러한 척도는 주어진

진술에 대한 동의나 의견 불일치의 강도를 나타낸다. ‘당신이 생각하기에 이 sw는 별론가, 평균

인가, 아님 좋다고 생각하는가? 와 같은 질문을 예로 들 수 있다. 이 차트에서 데이터들이 어떻게

사이에 간격이 있는 열로 분할되었고 모든 축이 라벨링되었으며, 차트는 또 어떻게 Figure 6.3

처럼 간단한 라벨에 부합하면서 유일하게 제목이 붙여지게 되었는지 주목해라.

연속적인 데이터의 경우 데이터가 구별되는 범주로 배열되지 않고 실제 값을 취할 수 있으므로

막대 차트의 대안이 필요하다. 이 경우 히스토그램이 사용된다. 히스토그램은 열이 특정 데이터

항목의 발생 빈도를 나타낸다는 점에서 막대 차트와 유사한 데이터 표현 방식을 가지고 있다.

하지만, 히스토그램은 연속적인 데이터를 나타내기 때문에, 고유한 범주로 나누는 방법을

당신에게 달려있다. 막대 차트는 자신이 나타내는 데이터내에 정의된 범주에 기초하여 그들을

위해 정의된 범주를 갖고 있음을 기억해라. 예시로, 그림 6.3에서 히스토그램은 컴퓨터 공부를 한

학생 100명의 나이를 보여주고 있다.

그림 6.3에서 졸업생들의 나이는 나이 범위 별 10개의 독특한 범주로 나뉘었다. 각 열은 정의된

범위에 속하는 학생의 수를 나타낸다. 이 막대 차트가 현재 어떻게 접촉하고 있는지 주목해라.

이는 데이터가 연속적이고 범주 간에 절대적으로 단절되지 않는다는 사실을 강조한다.

그 대신에 당신이 그 범주를 20 에서 25, 25 에서 30 등으로 정의할 수 없는 이유는 없다. 이

결과는 그림 6.4의 히스토그램에서 보여준다.

그림 6.4는 결과에 대한 더 번잡한 해석을 제공하며, 아마도, 졸업할 때 학생 연령의 확산에

대한 더 빈약한 개요를 제공한다. 이 그림은 연속 데이터를 히스토그램으로 표시하기 위해

적절한 범주를 신중하게 선택하는 것이 중요함을 강조한다. 데이터를 너무 많은 범주로

나누는 것은 큰 격차를 야기할 수 있는 반면, 데이터를 반대로 너무 적은 범주로 나누는 것은

기본 분포를 균등하게 나타내지 못하고 넓고 높은 막대를 만들어 내게 된다.

보고서에서 사용하고 싶어하는 또 다른 차트는 파이 그래프이다. 파이 그래프는 데이터 내의

범주의 비율을 나타내는 데 사용한다. 예를 들면, 그림 6.5는 그림 6.2에서 보여준 데이터와

동일한 데이터를 나타내고 있다. 그림 6.2는 실제 점수의 분포를 보여주는 반면, 그림 6.5의

파이 그래프는 특정 학위 분류를 가진 학생의 비율을 보여준다. 어떤 그래프를 쓰는가는

당신이 보고서 내에서 무엇을 강조하려하는지, 혹은 어떤 것을 설명하려하는지에 달려있다.

비율을 논하고 싶다면 파이 그래프를 사용하고 분산/분포에 대해 논하고 싶다면 막대 그래프를

이용할 것이다.

파이 그래프는 다양한 크기과 모양을 가진다. 3차원, exploded 형, 색깔, 그림자, 바퀴 등 여러

가지가 존재한다. 그리고 대부분의 스프레드 시트 패키지들이 이러한 형식들을 제공한다.

그래프를 어떻게 나타내느냐는 당신에게 달려있지만, 그래프 표현에 너무 집착하여 나타내려는

데이터의 진짜 의미를 모호하게 만들지 않도록 해야 한다.

여러 대상들의 비율을 비교하기 위해 여러 파이 그래프들을 사용할 수 있는 반면, 그림 6.6과

같은 혼합형 막대 차트는 이러한 비교를 더 분명하게 표현하는 데 사용할 수 있다. 이러한 경우,

4개 학교의 학위 성적 분포가 제시되어 있다. 이 차트에서 어떻게 범례가 포함되어 열에서

사용되는 음영을 식별하는지, 그리고 이 범례와 음영이 어떻게 1등급부터 불합격까지의 등급에

기초한 논리적 하향식 접근법을 따르는지 주목해라.

만약 실제 값을 제외하고 범주들 간의 비율을 알아보는 데 관심이 없다면, 그림 6.7과 같은 혼합

막대 차트를 사용할 수도 있다. 이 그림에서 당신은 성별에 따라 분류된 컴퓨터학과 졸업생의

수를 볼 수 있다. 막대들은 학위 분류 순서로 배열되어 있고 성별에 따라 나뉘어져 있다. 이러한

경우에 성별은 각 학위 분류에 의해 식별된다.

그림 6.7에 있는 막대 그래프가 오히려 더 통상적인 형식이다. 오늘날 이러한 그래프의 장점을

극대화하기 위해 3차원 구성, 색, 음영 등 여러 요소를 추가하는 것을 본 적이 많이 있을 것이다.

하지만, 전에도 얘기했듯이, 묘사하려하는 진짜 의미를 모호하게 해서는 안되며, 중요한 결과들을

정교한 다이어그램과 그림 자료 뒤에 숨겨버리는 경우를 만들어서는 안된다.

그림 6.8은 그림 6.7에서 사용된 데이터를 가져왔고 그룹 별로 재정렬을 하면서 그림 6.7의

결과와는 조금 다른 방식으로 표현을 하고 있다. 여기서 막대는 학위 분류로 나눠지고 이러한

등급은 성별로 그룹화되었다. 다시 한번, 이러한 데이터를 어떻게 표현하는지는 당신에게 달려

있으며 당신이 어떤 것을 강조하려는가에 따라 달라진다. 그림 6.7은 각각의 등급이 어떻게

남자와 여자 사이에서 어떻게 분포하는지 보여주는 것과 관련이 있다. 그림 6.8은 모든 남자에

대한 성적의 분포과 모든 여자에 대한 성적의 분포를 보여주는 것과 관련이 있다.

그림 6.7과 6.8의 한 가지 단점은 남자와 여자의 총 인원 수가 다르다는 것이다. 이상적으로

남성이 1등, 2:1 등을 획득하는 비율을 보고 싶어하며, 이를 여성이 동일한 등급을 획득하는

비율과 비교할 수 있다. 예를 들어, 평균적을 여성들이 남성들보다 더 많은 1등급을 받는가?

이를 증명하기 위해 데이터에 대한 통계적 분석이 필요할지라도, 그래프는 해당 상황에 대한

분명한 시각적 개요를 나타내줄 수 있다. 이 두 경우 모두 y축은 학생 수와 달리 백분율로

다시 표시되며 그래프는 성별에 따른 성적 분포와 더 나은 비교를 제공한다.

또 다른 형태의 유용한 그래프는 선 도표(line chart) 이다. 이러한 그래프는 일반적으로 시간에

따른 경향을 보여주는 데 사용된다. 그림 6.9는 1등급과 2:1이 1980년에서 1999년 사이 학위를

받은 경향을 보여주고 있다. 이는 이러한 데이터들의 시각적인 해석만을 제공한다는 것을

기억해야 한다. 실제로 정말로 중요한 경향인지 아닌지를 결정하기 위해 데이터들에 대한 통계적

분석을 수행해야 한다. 이러한 문제들을 처리하는 데 이용 가능한 수많은 텍스트들이 존재하기

때문에 통계적 테스트는 이 책의 범위를 벗어난다.

이 섹션에서 주목해야 할 인기 있는 형식의 그래프는 상관 도표(scatter diagram) 이다. 상관

도표는 두 변수 사이의 관계를 보여주는 데 사용된다. 그림 6.10은 30명 학생들의 과제 수행

시간에 대한 과제 등급을 표시한다. 차트가 이 두 변수 사이에 관계가 있을 수 있다는 것을

암시하며 어떻게 일반적인 상승 추세를 나타내는지에 주목해라. 비록 이 관계의 강도와 중요성이

통계적으로 계산될 수 있지만, 그래프는 이 관계에 대한 시각적 해석을 제공하며, 이는 아마 일부

통계적 계산보다 따르기 더 쉬울 것이다. 이는 당신이 획득한 데이터의 어떠한 엄격한 통계적

해석을 생략할 수 있다는 것이 아니라 당신의 보고서에 적절한 수치와 표를 사용하여 이러한

계산을 지원하고 그들의 표현 방법을 개선할 수 있음을 말한다.

[6.4.3] 검사 항목

Saunders는 보고서 내의 표와 그림들을 완성했을 때 관찰해야 할 것들의 검사 항목들에 대해

말한다. 다이어그램이나 표 모두를 위해 당신 스스로에게 다음과 같은 질문을 할 것을 권한다.

\* 간략하지만 기술적이고 분명한 제목을 가지고 있는가?

\* 측정 단위가 명확하게 제시되었는가?

\* 사용된 자료들의 출처가 분명하게 제시되었는가?

\* 축약어들을 설명하는 부분이 포함되었는가?

\* 샘플 크기는 명시했는가?

- 다이어그램을 위해

\* 축 명칭은 분명하게 제시되었는가?

\* 막대와 구성 요소들은 동일한 논리적 순서를 따르는가?

\* 더 조밀한 음영이 더 작은 영역에 사용되었는가?

\* 키나 범례가 포함되었는가?

- 테이블을 위해

\* 명확한 열과 행의 제목이 있는가?

\* 행과 열이 논리적인 순서를 따르는가?

[6.4.4] 일반적인 실수

보고서 자체를 위해 수치와 표를 포함시켜선 안된다. 수치와 표들은 당신이 본문에서 주장하고

있는 것을 지지하고 당신이 만들고 있는 도표적 형태, 데이터, 결과 및 해석을 명확히 하기 위해

있어야 한다. 이것이 수치와 표를 사용함에 있어 가장 흔한 첫 번째 실수를 하게끔 한다. 그림

6.11이 보고서에 가치를 추가하지 않고 파이 그래프를 표시하는 예이다. 이 경우에, 조사

대상자의 100%가 ‘그렇다’고 응답함에 따라 통상적으로 비율을 나타내는 파이 그래프의 사용은

불필요하며, 보고서 내용의 실제적 가치가 없기 때문에 군더더기를 붙인 것처럼 보이게 된다.

그래프를 사용할 때 흔히 하는 두번째 실수는 다른 그래프가 당신의 데이터를 훨씬 더 명확하게

보여주고 있을 때 그것들을 부적절하게 사용하는 것이다. 그림 6.12가 그 예를 보여주는데, 이

경우엔 추세가 초점이 아닌데 선 도표/꺾은 선 그래프(line chart)가 사용되고 있다. 학위 등급의

기본 분포 형태를 식별하는 데 관심이 있을 수 있지만, 이 경우 막대 그래프가 더 적합할 것이다.

보고서에 그래프를 포함시킬 때 흔히 하는 또 다른 실수는 크기를 잘못 조정하는 것이다. 종종

이것은 제시된 데이터의 진정한 의미를 감추기 위해 의도적을 행해지곤 한다. 어떤 경우엔

데이터가 의미하는 바를 잘 모를 때 우연히 행해지기도 한다.

그림 6.13과 6.14는 1980년에서 1999년 사이 한 대학 부서의 지출을 나타낸다. 두 그래프 모두

정확하게 동일한 데이터에 대해 나타내고 있지만, 다르게 보여진다. 그림 6.13은 해당 기간동안의

급격한 감소 추세를 보여주는 반면, 그림 6.14에선 이 추세에 관점을 두고 이 기간동안 아주

약간의 변화만 일어났음을 보여준다. 하지만 6.13이 지출 변화의 세부 사항들에 대해 잘 보여주고

있지만, 6.14는 일어난 상황에 대해 별 다른 정보를 제공해주지 못하고 있다. 6.14에선 자세히

들여다보지 않으면 해당 기간 내 지출의 변화가 거의 없는 것처럼 보이지만, 그렇지 않다는 것을

알고 있어야 한다. 이러한 두 그래프는 그래프 크기를 적절하게 잡는 것의 중요성을 강조한다.

당신은 숨기려는 것이 아닌 보여주려고 하는 것을 결정해야 하고, 그래프에 따른 적절한 크기를

잡아야 한다.

[6.4.5] 다양한 종류의 그래프

봤을 수도 있고 혹은 사용해보고 싶어하는 다소 생소한 차트들이 그림 6.15에서 6.18까지

나와있다. 그림 6.15는 3차원 막대 그래프이고, 건조한 2차원 막대 그래프 외형을 강조하는 데

사용된다. 이러한 차트들이 반드시 데이터 표현에 있어 어떠한 중요한 것들을 추가해주는 것은

아니지만, 그것들은 시각적으로는 확실한 효과를 제공한다.

그림 6.16은 폴라 그래프(polar chart)이고, 이는 비교할 만한 여러 요소들을 가지고 변수들을

비교하는 데 사용된다. 예를 들어, 폴라 차트의 각 가지(arm)는 특정 요소를 나타내고, 각 형태는

관심이 있어하는 변수를 나타낸다. 따라서 각 형상은 각 축에 제시된 각 요인에 대한 각 변수의

유사성과 차이를 나타내는 표시를 제공한다.

그림 6.17은 도넛 그래프(doughnut chart)이고, 이는 파이 그래프를 나타내는 또 다른 방법이다.

이 방식의 장점은 생성된 각 변수들 간의 비율 비교를 가능하도록 하기 위해 같은 그림 안에

여러 파이 그래프를 모아 구성할 수 있다는 것이다.

그림 6.18은 요인 분석 그림이고, 이는 요인 분석으로 나온 두 개의 결합된 요인의 결과를

나타낸다. 이는 이차원 상에서 수 많은 요인들에 따라 변수들이 어떻게 그룹화 되어 있는지

보여준다. 일부 변수는 이 다이어그램에서 서로 밀접하게 연관되어 보일 수 있지만, 다른 차원의

다른 요인을 보면 그렇지 않다는 것을 알 수 있다. 요인 분석은 컴퓨터와 통계 패키지를

사용해야지만 실제로 수행할 수 있는 통계 기법이다. 계산들을 손으로 직접 수행하기엔 너무

복잡하다.

[6.5] 자료 참조와 표절 피하기

\* 이전에 언급되지 않은 것은 아무것도 없다.

이를 염두에 두고 적절한 참고 자료를 바탕으로 보고서 내 제시하고자 하는 작업을 뒷받침하는

것이 중요하다. 특히 학부 프로젝트에서, 진행하려 하는 대부분의 작업들이 아마 이전 다른

사람들에 의해 다뤄지고 쓰여졌거나 혹은 논의되었던 것들일 것이다. 따라서, 보고서 내에서,

그리고 특히 문헌 검토에서 하는 모든 주장은 이전의 연구를 참조함으로써 정당화되어야 한다.

보고서 내에 다음에 대한 자료를 참조한다.

\* 표절을 피해라. 다시 말해서, 다른 사람들의 작품을 자신의 것처럼 보이게 하기 위해 참조없이

그들의 아이디어, 사고, 단어, 그림, 다이어그램, 결과 등을 제시해서는 안된다. 표절은 우연히

혹은 의도적으로 생길 수 있지만, 두 경우 모두 심각한 학문적 범죄로 여겨진다. 그렇기 때문에

문헌 조사를 광범위하게, 확실하게 수행하여 단순히 다른 사람의 작품을 따라하는 것이 되지

않도록 해야 한다.

\* 문서 내용을 식별하여 작품을 다른 공인된 출판물과 문맥에 비교해봐라. 이는 어떻게 다른

사람들의 작품을 보강하고 확장하는지 보여주고 당신의 작품이 공인된 학문 분야 내에 어떻게

위치해 있는지를 보여줌으로써 당신의 보고서를 강화시켜줄 것이다.

\* 뒷받침해주고 입증해라. 당신의 주장을 뒷받침하고 모든 것들을 입증해라. 만약 당신이

어떤 부분에 있어 지속적인 주장을 하고 있다면, 당신은 연구 결과나 다른 작가에 대한 참조를

통해 이러한 주장들을 뒷받침 할 수 있어야 한다.

\* 정보의 원천을 식별해라. 당신의 보고서를 읽는 사람들에게 그들이 당신의 보고서 주제에

대해 더 자세하게 공부하는 데 사용할 수 있거나 혹은 다른 작업에 사용할 있는 연관된 작업들의

목록을 포괄적으로 제시해줘라. 정보의 근원을 분명하게 명시해줌으로써 당신의 보고서를 읽는

사람들은 당신이 사용한 기사들을 찾아낼 수 있을 것이다.

참조에 대한 두 가지 측면이 존재한다. 첫번째는 보고서 내에 참조를 올바르게 사용하는

방법이다. 즉, 인용에 대한 제시하는 방법과 형식이나 적절성 등을 말하는 것이다. 두번째는

보고서 끝에 이러한 참조을 올바르게 표시하는 방법이다. 두가지 측면은 차례대로 처리될 것이다.

[6.5.1] 참조 인용

일반적으로 말해서 참조를 인용하는 방법으로, Havard 시스템과 numeric 시스템 두가지가

존재한다. Havard 시스템은 사용하기에 더 좋은 시스템이다. 왜냐하면 numeric 시스템에선 각

참조가 어떤 유일한 숫자에 의해 식별되어야하고 이러한 것들이 당신이 보고서 내 참조를

추가하거나 삭제하기로 결정할 때마다 갱신되어야하기 때문이다. 종종 숫자 시스템은 또한

당신이 언급하고 있는 저자에 대한 어떠한 표시도 주지 않고 독자는 보고서 뒤에 있는 참조

목록을 통해서 검색해야만 이 정보를 찾을 수 있다. 오늘날, 많은 워드 프로세서 패키지들이

보고서 내의 참조들을 쉽고 빠르게 유지하고 갱신할 수 있도록 해주는 참조 관리 시스템을

지니고 있다. 4장에서 참조를 관리하는 데 도움을 줄 수 있는 수 많은 소프트웨어 도구들이

나열되었다. 하지만, 그러한 시스템을 가졌던 안 가졌든, Havard 스타일의 참조 시스템을

사용하도록 권장된다. 그리고 이 시스템이 숫자 시스템보다 훨씬 유연성이 좋고 명확하다.

Havard 시스템은 보고서 내에서 각 참조를 유일하게 식별할 수 있게 하기 위해 저자의 이름과

출판 연도를 사용한다. 예를 들어, 학부 프로젝트 보고서에서 나온 다음의 발췌문을 살펴봐라

-> case1) 컴퓨팅은 과학이 아니라 예술이라고 여겨진다. (Smith and Jones 1993: 20)

-> case2 ) 컴퓨팅은 과학이 아니라 예술이라고 여겨진다. 이는 그들의 문제를 정당화한

Smith 와 Jone에 의해 처음 제안되었다. (1993: 20)

Smith와 Jones에 의한 기사는 출판 연도에 의해 식별된다. 만약 동일한 연도의 여러 기사들을

참조하려 한다며, 각 기사를 유일하게 구별하기 위해 날짜까지 추가하게 될 것이다. 페이지

번호인 (20) 또한 식별된다. 이는 책을 참조할 때 흔히 볼 수 있는데, 그것은 분명히 많은

페이지를 가지고 있지만, 저널 기사를 참조할 때는 그렇지 않다.

동일한 참조에 의한 뒷받침되는 이 주장을 제시하는 대안적 방법은 다음과 같다.

-> Smith and Jone (1993: 20) 는 ‘컴퓨팅은 과학이나 공학보다 예술과 같은 삶에서 더 좋아

보인다.’

이전의 예시에서 Smith와 Jones의 생각은 우리의 말에 넣어져 있었기 때문에 인용 부호는

불필요했다. 하지만, 그들이 텍스트는 이제 그들의 말 그대로 사용되었기 때문에, 이 텍스트는

따옴표 안에 포함되어야만 우리가 아니라 그들의 말이라는 것을 보여줄 수 있다. 큰 텍스트

블록을 인용한다면, 어떤 식으로든 텍스트에서 두드러져 보이게 하면서 인용 부호없이 표시하는

것이 허용된다. 예를 들어, 기울임꼴 등의 다른 글꼴로 표시하거나, 넓은 여백 사이에 블록함

포함시켜 부각되게 하는 등의 방법을 이용할 것이다.

Cornford 와Smithson에 의하면, 다른 사람의 작업을 말 그대로 인용해야 하는 세가지 경우가

다음과 같이 존재한다고 한다. .

\* 원 저자가 당신이 할 수 있는 것보다 무언가를 더 간결하고 우아하고 분명하게 제시한 경우

\* 그 글을 쓴 사람이 특정 작가라는 것을 증명해야 하거나, 그것을 분석하기 위해 몇가지 글을

소개해야 할 필요가 있는 경우

\* 인용 목록과 같이 다른 말로 바꾸어 표현할 방법이 없는 경우

너무 많은 인용문과 다른 작가들에 대한 언급은 당신의 작품에서 어떠한 권위를 내주게 될 수

있다는 것을 기억하고 있어야 한다. 다시 말해서, 다른 작가들의 자료를 너무 많이 포함시켜

독자들 입장에선 인용된 자료들이 대부분 다른 이의 것이므로 작품에 대한 당신의 기여도를

식별하기 어려워질 수 있다는 것이다.

많은 기사들이 개인들에 의해 작성된다. Havard 시스템은 개인 저자에게 다음과 같이 맞춰준다.

-> 컴퓨팅은 과학이 아니라 예술이라고 여겨진다. (Johnson 1992)

하지만, 동일한 기사에 관련된 저자가 두 명 이상 있을 때, 당신은 일반적으로 첫번째 저자의

이름을 제외한 모든 것을 생략하고 대신에 다음과 같이 et al. 과 같은 문자를 사용한다.

-> 컴퓨팅은 과학이 아니라 예술이라고 여겨진다. (Johnson et al. 1992)

당신의 주장을 뒷받침하기 위해 하나 이상의 참조를 하고 싶을 때, 그것들을 알파벳 순서대로

포함시켜야 한다.

-> 컴퓨팅은 과학이 아니라 예술이라고 여겨진다. (Johnson 1992; Pter et al. 1995; Smith 1993:20)

당신은 또 다른 작가에 인용된 읽어 본 적없는 참조(2차적 참조)를 제시하고 싶을 수도 있다.

이러한 경우엔 읽은 기사만 나열하면 되고 참고 문헌은 다음과 같이 인용해야 한다.

-> 컴퓨팅은 과학이 아니라 예술이라고 여겨진다. (Johnson 1992, cited by Markos et al. 1996)

완전성을 위해 숫자를 참조하는 형식을 간략히 살펴보면, 이 경우에 각 참조는 고유한 숫자로

식별된다.

-> 컴퓨팅은 과학이 아니라 예술이라고 여겨진다 [1,2].

혹은

-> 컴퓨팅은 과학이 아니라 예술이라고 여겨진다. 

혹은

-> Smith and Jone [2] 는 ‘컴퓨팅은 과학이나 공학보다 예술과 같은 삶에서 더 좋아

보인다.’

이러한 경우에, 보고서 내에서 새로운 참조가 사용될 때마다 새로운 참조 번호가 부여된다. 그런

다음 보고서 뒷면에 알파벳 순서가 아닌 숫자 순서대로 참조가 나열된다. Smith와 Jone에 대한

참조를 삭제하기로 한 경우 삭제된 참조를 대체하려면 이후의 모든 참조 번호를 모두 다시

매겨야 한다는 점에 유의해야 한다. 이와 유사하게 보고서에 새 참조를 삽입하려면 모든 후속

참조 번호를 증가시켜야 한다. 이러한 상황에서 워드 프로세서에 참조 관리 시스템이 없을 경우,

검색 및 교체에오랜 시간이 걸릴 수 있다. 이러한 이유로 만약 선택권이 있다면, 가능한 한

자료를 참조하기 위해서 Harvard 시스템을 사용하는 것이 좋다.

참조 시 사용할 수 있는 세가지 약어는 다음과 같다

\* op. cit. -> 이미 인용된 작품에서

\* ibid. -> 동일 장소에서

\* loc. cit. -> 인용된 장소에서

이러한 약어들이 짧은 기사들에서 발견되고 숫자 참조 시스템에서 사용될 때 유용하지만,

좀 더 긴 문서들에서 조심히, 적절히 사용되어야 한다. 이러한 약어는 보고서의 동일한

페이지 또는 최대 한 페이에서 발생하는 참조로 제한하도록 해야한다. 사전에 미리 두 페이지

이상에서 발생한 참조를 연결하려면 독자가 따라하기 어렵다.

위에서, 보고서 내에서 참조한 자료로부터 정확이 어떤 기사를 참조하고 있는지 확인하고

동일한 벙법으로 두 기사를 식별하지 않도록 해야한다. 일관되게 한 가지 스타일의 방법을

사용해야 한다. Harvard 시스템에서 숫자 시스템으로, 혹은 숫자 시스템에서 Harvard 시스템으로

전환시키지 말고 또한 두 가지 형식을 혼합하여 사용하지도 말아야 한다.

[6.5.2] 참조 목록

일반적으로, 사용한 모든 참조들을 나열할 가장 좋은 위치는 각 장 끝에 있는 각주나 목록

하단에 있는 각주와는 반대로 보고서의 마지막 끝 부분이다. 이것은 독자에게 쉽게 접근할 수

있는 모든 관련 자료의 단일 개요서를 제공한다. 당신이 사용한 기사들은 어느 한 참고 문헌이나

참고 제목 아래에 제시된다. 참고 문헌은 보고서 자체 내에서 업급된 기사만 나열한다. 참고

문헌에는 프로젝트에서 사용한 모든 기사가 나열되지만 보고서 본문에서 반드시 언급되지는

않는다. 참고 문헌은 당신의 연구를 진전시키거나 보다 심층적으로 이해하는 데 관련된 모든

자료들을 확인한다는 점에서 독자들에게 유용하다. 학부 프로젝트와 책의 경우 참고 문헌을

포함하는 것이 더 적절할 수 있지만, 대학원 프로젝트의 경우엔 그렇지 않을 것이다. 지도 교수가

어떠한 접근법을 사용해야 하는 지 조언할 수 있어야 한다.

참조 사항들을 나타내는 방법은 당신이 사용하는 참조 시스템인 Harvard나 숫자 시스템에 따라

달라진다. 오직 Harvard 시스템이 기본적으로 숫자 시스템과 동일하기 때문에 더 자세히 논의될

것이다. 숫자 시스템과의 유일한 차이는 숫자 시스템이 숫자 식별자를 가장 먼저 나타내고 또

이러한 숫자 순서대로 제시된다는 것이다.

Harvard 시스템에서 기울임꼴, 콤마, 콜론, 대문자, 약어, 그리고 괄호의 사용은 당신이 속한

기관 고유의 스타일에 따라 달라진다. 그러나 Harvard 참고 문헌은 연대순으로 제시된 작가의

글과 함께 알파벳 순서대로 제시되어야 한다. 다음은 참고 문헌 종류에 따른 예시는 다음과 같다.

\* 서적

제시된 정보로부터 출판사가 명확하게 알려진 한 출판사 이름에 부가적인 용어를 포함시킬

필요는 없다. 제시된 날짜는 그 책의 판본이 처음 출팔된 날짜를 나타낸다. 이것은 책의 연대를

나타내는데 이것은 몇 년 후가 될 수 있는 재인쇄 날짜를 참조한다고 날짜를 참조한다고 해서

분명하지 않을 것이다(갱신 날짜가 아닌 첫 출판일을 제대로 찾아봐야 한다).

\* 저널 기사

동일한 해에 같은 저자에 의해 생산된 두 기사를 각각 고유하게 식별하기 위한 문자 사용에

유이해야한다. Ex) 2019a / 2019b

\* 웹 주소

인터넷 사이트에 대한 참조는 반드시 http를 포함해서 웹 사이트 전체 주소를 포함시켜야 한다.

적절한 곳에 페이지 제목, 기사, 그리고 사이트 이름을 명시해야 한다. 이러한 참조들은 사이트에

접속한 날짜 또한 확실하게 포함시켜야 한다. 왜냐하면 인터넷 자료는 계속해서 바뀌고, 이러한

참고 자료들은 매우 빠르게 구식이 되기 때문이다.

\* 회사 간행물

\* 논문

\* TV 프로그램

위에 제시된 참고 문헌들은 결코 포괄적이지 않으며, 당신은 의심의 여지없이 이러한 예시들에

의해 다루지지 않은 불명확한 출처로부터 기사, 일부 데이터 또는 여러 자료들을 접하게 될

것이다. 그러나, 당신이 속한 기관이 어떠한 구체적인 가이드 라인을 가지고 있지 않는 한,

당신은 당신이 생각하기에 적절하다고 생각하는 방식으로 참조를 제시해야 할 것이다. 지도

교수가 당신을 도와줄 수 없다면, 두 가지만 딱 기억해라. 첫번째로, 참고 문헌은 충분히

명확해서 보고서를 읽는 모든 사람들이 당신이 어떠한 기사에 대해 언급하고 있는 것인지를

알 수 있도록 해야 한다. 두번째로, 독자가 원한다면, 당신이 언급한 기사를 찾기 쉽도록 그에

대한 충분한 정보를 제공해줄 수 있어야 한다.

[6.6] 소프트웨어 문서화

[6.6.1] 소개

프로그램 코드, 시스템 분석 및 설계 노트, 그림 및 시스템 설명서의 내부 주석부터 테스트 계획

및 사용자 가이드에 이르기까지 소프트웨어 한 개를 지원하는 데 필요한 문서는 엄청 많을 수

있으며 광범위한 문제를 다룬다. 다음의 목록은 사용자가 제작하는 소프트웨어를 지원하기 위해

프로젝트에 포함시키고 포함할 수 있는 주제와 문서를 항목화한다.

\* 소개/개요 : 프로그램에 대한 단순한 소개, 무엇을 하는지, 누가 하는지

\* 채택한 기술적 솔루션 : 어떤 기술적 해결책이 구현되었는지, 이상적인지 아닌지, 대안이

존재하는지 존재하지 않는지

\* 디자인 : 시스템 분석, 시스템 디자인, 인적 요인, 스토리 보드 등

\* 소프트웨워 엔지니어링 정보 : 프로그램 디자인, 구조, 정의 언어, 테스트 계획 등

\* 사용한 개발 방법 : 진화적 출시 모형 등

\* 발생한 문제 : 버그, 에러, 불완전 코드 등

\* 한계 : 프로그램에 존재하는 한계. 예를 들어 이 프로그램은 특정 사이즈의 파일만을 다룰 수

있다, 계산 정확도가 10%에 불과하다 등

\* 프로그램에 필요한 SW / HW 자원

\* 다음 단계 : 만약 계속한다면, 혹은 다른 사람이 당신의 프로젝트를 넘겨 받아 수행한다면,

소프트웨어의 어떤 부분이 다음으로 개발되어야 하는가? 프로그램의 어떤 부분이 새로운

특성으로 강조될 수 있는가? 코드, 문서, 코멘트 등이 미래에 누군가가 당신에게서 쉽게 념겨

받을 수 있는 수준에 있는가?

\* 소프트웨어 평가 : 의도한 대로 잘 진행되었는가? 사용자의 요구 사항을 만족시키는가?

\* 사용자 가이드 : 의도한 사용자를 위해 적절한 수준에서 세부적으로 작성되어야 한다.

당신의 프로젝트 성격에 따라, 이러한 각 영역에 대해 더 자세히 혹은 덜 자세히 나타내야

할 것이다. 디자인, 분석, 테스트 계획 등과 같은 문서화를 완성하는 방법은 개발 과정, 채택된

방법, 그리고 당신이 수행한 프로젝트의 유형에 따라 달라지기 때문에 이 책의 범위를 벗어난다.

예를 들어, 순수 프로그래밍 프로젝트는 포괄적인 분석 다이어그램, 테스트 계획, 그리고 시스템

문서화를 필요로 하는 반면, 단순하게 일부 코드를 개발하는 프로젝트는 그렇지 않다. 여기서의

초점은 프로그램 주석 작성과 사용자 가이드를 작성하는 데 맞추고 있다. 왜냐하면, 이러한

것들이 당신이 개발하는 코드의 일부에 반드시 포함되어야 하기 때문이다.

[6.6.2] 주석 프로그램 코드

프로그램 주석 코드는 사용하는 프로그래밍 언어, 개발되는 코드 스타일, 그리고 당신이 속한

과정이나 프로젝트의 요구사항에 따라 달라진다. 당신의 코드는 설명할 때 따라야 할 일반적인

가이드라인이 존재한다.

\* 당신이 작성하려하는 프로그램의 목적을 이해해야 한다. 누가 사용할 것이고, 누가 유지하거나

개선시킬 것이며, 그리고 누가 그것을 표시할 것인가? 지식 수준은 어느 정도인가? 단지 몇 가지

아이디어를 테스트 하기 위한 개인적 용도로 조그마한 프로그램을 만드는 것이라면, 다른

사람들이 사용할 프로그램만큼 많은 설명을 달 필요가 없을 것이다.

\* 프로그램 내 적당한 수준의 주석을 달아야 한다. 너무 많이 달지도, 너무 적게 달지도 말아야

하며, 매 줄마다 주석을 다는 것은 피해야 한다. 주석은 프로그래머에게 코드 상에서 불분명한

요소들에 대해 말해주어야 하며, 사용되는 프로그래밍 언어에 대해 설명하기 위해 존재해서는

안된다. 프로그램에 적합한 변수 이름과 논리 구조를 충분히 사용했다면 주석을 제한되어야 한다.

즉, 충분히 설명된 코드에 대해선 불필요한 주석을 피해야 한다.

\* 각 기능, 절차, 객체, 블록, 화면 등에 주석을 다는 것은 바람직하다. 이는 최소한 프로그램의

각 주요 구성 요소가 무엇을 하는지 설명해줄 것이며, 누군가가 프로그램이 어떻게 동작하고

구조화되는지 이해하기 위해 요구하는 것일 수 있다.

\* 프로그램 내 여러 복잡한 코드들 사이에 묻히지 않도록 주석들을 돋보이게 만들어야 한다.

\* 장황하나 설명은 피하고 주석은 반드시 간략하지만 분명하게 작성해야 한다.

\* 주석을 꾸미는 데 시간을 낭비하지 말야야 한다. 주석은 단지 코드에 대한 이해와 설명을

위한 요소일 뿐 예쁘게 보이기 위해 존재하는 것이 아니다.

\* 프로그램 시작부에 개발자 이름, 날짜 버전 정보, 프로그램에 대한 설명 등 필수 정보들을

간략하게 주석으로 작성해야 한다. 이러한 주석은 보통 블록 형식으로 묶어 작성한다.

\* 소프트웨어를 개발하고 수정할 때, 주석 또한 유지하거나 갱신할 수 있도록 해야 한다. 훨씬

이전의 버전과 다른 버전의 프로그램을 참조하는 오래된 코드는 의미가 없다.

요약하면, 주석의 수준과 스타일에 관해서는 지도 교수의 가이드 라인을 따르는 것이 가장 좋은

방법일 것이다. 또한 부서 내의 가이드 라인이 존재하여 해당 부서만의 스타일이 존재한다면,

그를 따라야 한다.

[6.6.3] 사용자 가이드 작성

최근 몇 년간 사용자 가이드에 대한 많은 연구가 있었다. 사용자 가이드 구조, 표현과 내용,

사용성, 훈련성, 미니멀리스트 작업 문제 등 이 책의 의도된 범위를 벗어나는 많은 것들에 대한

연구가 있었다. 이러한 경우, 일반 대중이 가이드를 사용하는 것이 아니라 컴퓨터 프로젝트나

프로젝트 평가의 일부로 자체 기관 내에서 사용한다는 점에서 더 좁은 관점에서부터 사용자

가이드에 대한 관심을 가질 가능성이 높다.

이러한 맥락에서, 당신이 개발하는 모든 사용자 가이드는 최종 보고서에 분리된 문서 형식으로

제시되거나 혹은 부록 내에 포함될 가능성이 높다. 사용자 가이드를 어떻게 표현할 것인가는

당신에게 달려있지만, 가이드 내용이 길면 길수록 그것들은 별도의 문서로 표시하는 것이 더

현명할 것이다. 필요한 문서 범위에 대해 지도 교수의 조언을 구해야 한다. 어떤 경우든, 사용자

가이드는 최소한 다음과 같은 내용의 정보 정도는 사용자에게 제공해야 한다.

\* 소프트웨어 개요 : 무엇을 하는지, 누구를 위해 만든 것인지(사용자가 누군지)

\* 하드웨어 요구 사항 : 메모리 요구 사항, 디스크 용량 등

\* 소프트웨어 적재 및 설치 방법

\* 소프트웨어 실행 방법

\* 소프트웨어 종료 및 삭제/제거 방법

\* 알려진 세부적인 문제 사항 및 프로젝트 제약 사항

더 넓은 범위에서 말하자면, Rogerson에 따르면 사용자 매뉴얼은 다음과 같은 세가지

목표들을 만족시켜야 한다.

\* 도움이 필요할 때 소프트웨어에 대한 실질적/실용적 정보를 제공한다.

\* 처음 이용하는 사용자가 최소한 어려움없이 빠르게 시작할 수 있도록 도움을 준다.

\* 경험있는 사용자들은 신속하게 생산성을 발휘할 수 있도록 도움을 준다.

프로젝트의 일부로 사용자 가이드를 작성할 때, 목표 대상의 사용자를 식별하는 것부터

시작해야 한다. 초보자가 소프트웨어를 이해할 수 있는 포괄적인 가이드가 필요한가? 아니면

해당 기능에 대한 간단한 개요만으도 충분한가?

사용자 매뉴얼에는 보통 두가지 형식이 존재한다. 첫번째는 교육 메뉴얼(training manual)로,

사용자들은 서로에 기반을 둔 여러 가지 예시들을 통해 소프트웨어 사용법을 배운다. 두번째는

참조 메뉴얼(reference manual)로, 숙련된 사용자는 참조 매뉴얼을 통해 프로그램의 특정 기능에

대한 설명을 위해 적절한 지점에서 매뉴얼을 이용할 수 있다. 문서 구성 방법은 주로 의도한

사용자를 기반으로 하기 때문에 숙련된 사용자의 경우 참조 설명서만 필요할 수 있다. 그러나

경험이 부족한 사용자에게는 진화적인 예시가 더 적합할 수 있다. 게다가, 사용자의 성격에 따라

다른 이름으로 저장, 페이지 설정 등과 같은 간단한 작동 원리를 설명하는 자세한 설명을

제공해야 할 수도 있다. 또한 사용자 가이드에서 덤프를 검색하여 사용자가 예상한 대로

결과가 나타나지 않을 때 사용자가 가이드를 올바르게 따르고 있는지 알 수 있게 해주는

것이 좋다. 발생할 수 있는 오류나 이를 해결할 수 있는 방법에 대한 설명 또한 추가해주는

것이 좋다.

[6.7] 요약

\* 보고서를 작성하려할 때, 독자들을 먼저 고려하고, 보고서 길이에 대한 모든 제한을 인식할 수

있어야 한다. 보고서를 구조화하기 위해 하향식 접근법을 사용하고 보고서 내 모든 섹션들이

작업이 진행됨에 따라 변화하는 것을 허용해야 한다. 보고서를 작성하는 특정 순서가 존재하고

구성해야 하는 구체적인 방법들이 있다. 작성 스타일을 연습하고 개선시킬 수 있는 방법을

찾아봐야 한다.

\* 요약은 맨 마지막에 작성해야 할 것들 중 하나이다. 이는 간결하고 분명해야 하며, 보고서의

맥락, 범위, 기여도 등을 요약해야 한다. 요약을 컨텐츠 목록으로 표시해서는 안된다.

\* 차트와 그래프들은 보고서 내용과 외형은 강조할 수 있다. 이것들은 적절하게 사용되어야

하고 각각은 유일하게 라벨링되고 제목이 붙여져야 한다. 또한 당신은 당신이 나타내려는 요점을

분명하게 하기 위해 올바르게 조정해야 한다.

\* Harvard 시스템은 보고서 내 참고 자료들을 위해 사용할 수 있는 가장 적합한 시스템이다. 각

기사들은 유일하게 식별될 수 있어야 하고, 각 참조들은 잘 갖춰져서 독자들이 참고된 해당

기사들을 추적할 수 있어야 한다.

\* 소프트웨어 문서화는 프로그램 주석 작성부터 사용자 가이드 작성까지 다양한 주제를 다룬다.

이 장에서는 일반적으로 교육 매뉴얼 또는 참조 매뉴얼로 제시되는 사용자 가이드의 개발에

초점을 맞추었다.