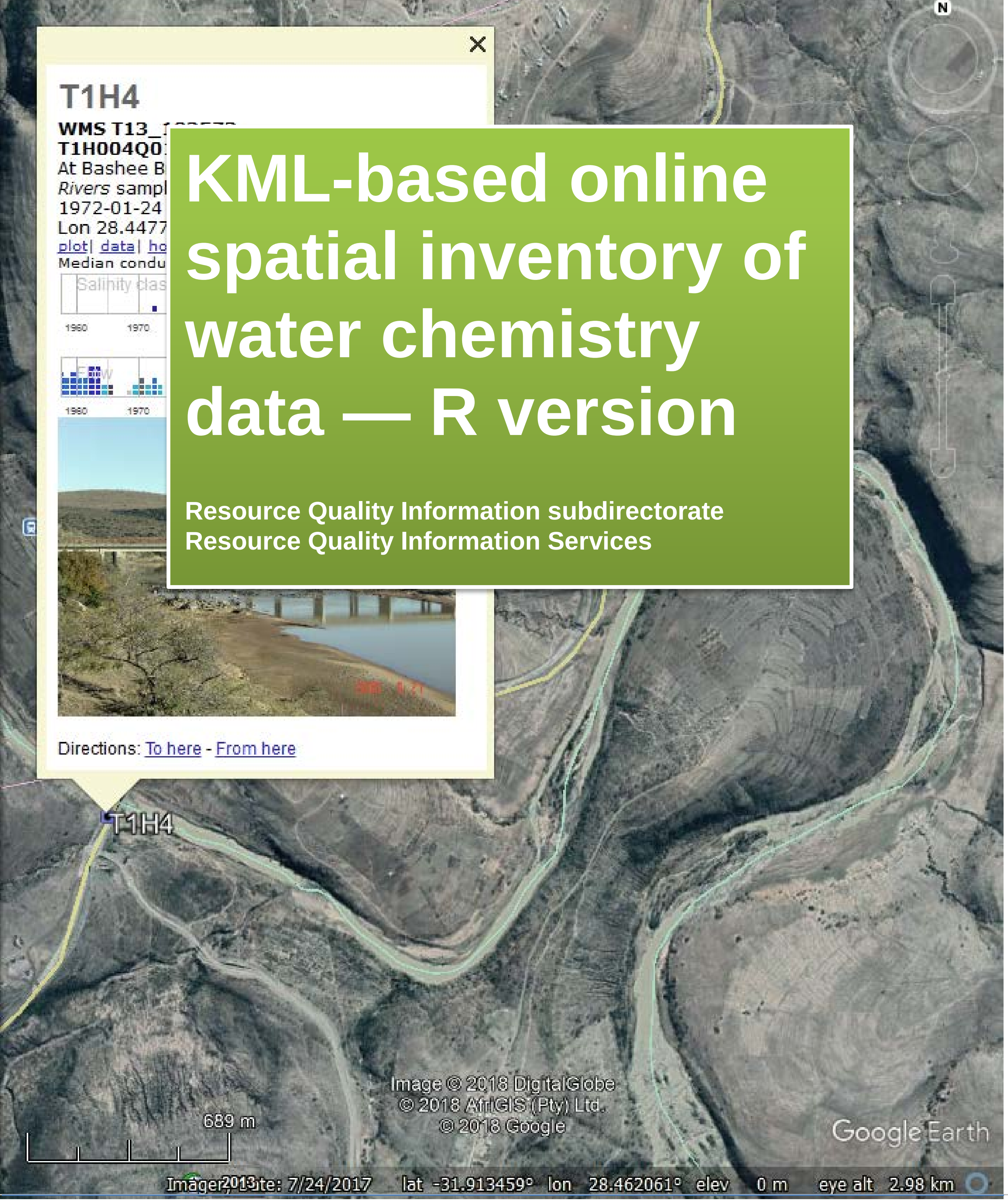

KML-based online spatial inventory of water chemistry data — R version

Resource Quality Information subdirectorate
Resource Quality Information Services

water & sanitation
Department:
Water and Sanitation
REPUBLIC OF SOUTH AFRICA

DOCUMENT STATUS

WORKING TITLE:	KML-based online spatial inventory of water chemistry data — R version (Conversion to R of the application that creates the online spatial inventory of water chemistry data)
AUTHOR:	Michael Silberbauer
REPORT STATUS:	**Third draft**
RQS REPORT NUMBER:	N/0000/00/RD1/2011
DATE:	December 2018

Executive summary

When Google Earth became publicly available in 2005, it provided the ideal platform for an interactive inventory of the Water Management System (WMS), which is the South African national water quality database. The first working prototype of a Google Earth driven spatial inventory used a set of ArcInfo-AML, awk and DOS scripts for converting data from the WMS Informix data base to flat text files, processing these to produce the final set of data plots, data listings and Keyhole Markup Language spatially-referenced XML (KML) files for Google Earth (Silberbauer and Geldenhuys, 2008 and 2009).

In 2011, the Unix-based ArcInfo scripts were becoming obsolete, so the software was ported to R (R Core Team, 2018). The R package has a wide range of statistical functions and graphics abilities, including geographical representation of data. R provides many opportunities for further development on a variety of platforms, including web-based applications. The package is open-source, extensible and licensing is free.

The main components of the Google Earth inventory are the KML files themselves – containing locality, time and attribute information – the multivariable time-series plots, the data listings and the Maucha multivariable point ionic symbols. This report describes the purpose, structure and operation of the application that creates the KML monitoring site files, `kml_WMS_mnpts_html.R`. This R script includes 20 functions that deal with formatting text, and creating each KML file with its associated HTML web file. A key component of the inventory is the popup information box that displays attribute information when the user clicks on a site, and which provides links to further information, such as time series plots, source data and the Department of Water and Sanitation's hydrological database.

Functions	
fn	function (lgp, fty, reg, msym, chem.333)
GRPhead	function (lgp, pr, msym, regiontype, chem.333)
GRPtail	function (lgp, pr, msym, chem.333)
HTMhead	function (pr, spr, regiontype, chem.333, txt.htp, txt.url, txt.hri)
HTMtail	function (pr, chem.333)
ico	function ()
KMLhead	function (pr, spr, regiontype, msym, chem.333, txt.htp, txt.url, txt.hri)
KMLtail	function (pr, msym, chem.333)
lgpIcon	function (lgp, n)
Located_On_Group	function (s)
mkey	function (f, txt.htp, txt.url, txt.hri)
mtxt	function (msym)
SECbody	function (pr, lgp, sse, channel, msym, chem.333)
SEChead	function (pr, lgp, se, sse, msym, chem.333)
SECtail	function (pr, se, lgp, msym, chem.333)
tokml	function (regiontype, s, msym, chem.333, txt.htp, txt.url, txt.hri)
ver	function ()
wmaname	function ()
WMSCheck	function (queryresult)

Introduction

In 2005, Google released a free online version of the Keyhole Earth Viewer package under the name Google Earth™. This software uses advanced techniques to speed up the laborious process of combining different sources of spatial data and rendering them in a perspective, 3D view of the Earth's surface. Real-time landscape fly-throughs become practical on home computers (Crampton 2008). The most important component of the system for practical applications is the extensible markup language, called Keyhole Markup Language or KML (Google 2018). This allows users to simply and quickly place their own spatial data on the Google Earth landscape without the need for advanced programming. Google has made many enhancements to the software, such as the ability to add time tags to user data and to splay out overlapping points. Many new base datasets have become available and the resolution of the satellite data acquired by Google is good enough for the identification of structures such as weirs and sewage works in many areas. Streetview data became available for South Africa in 2010 and provided an even closer perspective of the situation on the ground. Streetview images from bridges provide an invaluable perspective of the state of rivers and riparian habitat at the time of the photograph. Built-in point-to-point driving directions are of great assistance when planning fieldwork.

This report describes the `kml_WMS_mnpts_html.R` script in detail. The purpose of the report is twofold: to aid in maintenance of the script and to provide nuts-and-bolts information for anyone wanting to port the process to another platform.

Methods

The software consists of a main routine that generates eight groups of files (Table 1), plus several functions handling repetitive processes or simply making the code easier to understand and maintain. The version discussed here is 6.7.

Table 1. The eight types of KML inventory file created by the spatial inventory system.

	Surface water sites		Groundwater sites	
	Primary drainage regions	Water management areas	Primary drainage regions	Water management areas
Marker symbols	A to X, + 'O', 'Y', 'Z'	1 to 19	A to X, + 'Y', 'Z'	1 to 19
Maucha symbols	A to X, + 'O', 'Y', 'Z'	1 to 19	A to X, + 'Y', 'Z'	1 to 19

The software environment chosen is primarily the R statistics system (R Core Team, 2018) with the database package `RODBC`. The advantages of R are that it has no licence fee, has a large and helpful user community, provides a range of statistical analyses and has excellent graphics capabilities. Developers familiar with other programming and scripting languages might wish to achieve the same end results using software that they are more familiar with.

Formerly, an ArcInfo AML script converted a dBase inventory file from the Water Management System to a list of sites with information on drainage regions and management areas. An R script, `wms2nms.R` replaced this process in 2012. This script performs geographical point-in-polygon checks to ensure that each monitoring site is in the correct drainage region and water management area, and outputs a shapefile and a text list of site information.

The Department of Water and Sanitation combined the original 19 water management areas into 9 larger areas in about 2012. This inventory still uses the 19 water management areas because the 9 new areas are larger, with many more sites, which would cause memory problems on computers and portable devices.

The `kml_WMS_mnpts_html.R` script needs to run on a computer that has an ODBC connection to the WMS database. The connection should be read-only, to avoid any unfortunate accidents. The setup requires the installation of Informix-Connect and its configuration to connect to the database server.

Main routine

```
library(foreign)
library(RODBC)
library(RCurl)
library(httr)
odbcCloseAll()
if (!exists ("monitoringSites", mode="function") ) source
("C:/data/program/R/monitoringSites.R")

# check that the method for looking up weir photos is working - sometimes gives false
negative on slow line:
set_config(use_proxy(url="rqswtmg101.dwa.gov.za",port=8080))
img.hispic <- "http://www.dwa.gov.za/hydrology/Verified/CGI-
BIN/HIS/Photos/A2H055.JPG"
if(http_error(img.hispic, followlocation = 0L, USE.NAMES = FALSE)) stop("Check
Internet link and proxy")

options(stringsAsFactors = FALSE) # not working with factors so would rather have
strings
time.start <- as.POSIXct(format(Sys.time(), "%Y-%m-%d %H:%M:%S"))
chem.333 <- FALSE # switch is TRUE to produce output for the top 333 sites only
# use variables for departmental URL construction to make updates easier
txt.htp <- "http://"
txt.url <- "www.dwa.gov.za/"
txt.hri <- "iwqs/"

s <- monitoringSites()
s <- subset(s, s$No > 0)
s$ID[s$ID == ""] <- NA # is this wise? or even necessary?
s$WMANUM[s$WMANUM == ""] <- 0 # missing WMA number
s$WMANUM <- as.numeric(s$WMANUM)
s$WMANAME[s$WMANAME == ""] <- "WMA_unknown" # missing WMA name
s <- within (s,
             Located_On_Type <- ifelse(!is.na(Located_On_Type), Located_On_Type,
"Class pending")
             # Thanks to Joris Meys, http://stackoverflow.com/questions/7488068/test-
for-na-and-select-values-based-on-result
)

if(chem.333) {
  # read the sites from the WMS:
  channel<-odbcConnect("wmsdb")
  wmstables<-sqlTables(channel)

  q <- paste("SELECT UNIQUE mon_feature_id ",
             " FROM programme_sample ",
             " WHERE mon_program_id = 146")
  programme.sites <- sqlQuery(channel, q)

  print("*** Only processing \"CHEM 333\" sites ***")
  #   c.333 <- read.csv("C:/data_large/av/WMS/NCMP_Inventory_2012-05-21.csv")
  #   s <- s[s$PointID %in% c.333$Monitoring_Point_ID, ]
  s <- s[s$PointID %in% programme.sites$mon_feature_id, ]
}
msym <- FALSE

tokml("PDG", s, msym, chem.333, txt.htp, txt.url, txt.hri) # primary drainage
regions, no Maucha

if(!chem.333)
  tokml("WMA", s, msym, chem.333, txt.htp, txt.url, txt.hri) # water management
areas, no Maucha

msym <- TRUE

if(!chem.333) for (regiontype in c("PDG", "WMA")) {
  # primary drainage regions or water management areas, with Maucha option
  tokml(regiontype, s, msym, chem.333, txt.htp, txt.url, txt.hri)
}
odbcCloseAll()
time.end <- as.POSIXct(format(Sys.time(), "%Y-%m-%d %H:%M:%S"))
print(paste0("Started: ", time.start) )
print(paste0("Finished: ", time.end, " (", time.end - time.start, ")") )
print(paste("chem.333 =", chem.333))
```

The main routine runs the function `msym` once each for primary drainage regions (PDG), water management areas (WMA), first without and then with Maucha ionic diagrams as symbol markers. It runs a separate function for reading monitoring sites prepared by `wms2nms.R`.

```
s <- monitoringSites()
s <- subset(s, s$No > 0)
```

The main routine also has a seldom-used option for updating only the national chemical monitoring programme sites:

```
chem.333 <- TRUE.
```

Main controlling function

```
tokml <- function(regiontype, s, msym, chem.333, txt.htp, txt.url, txt.hri) { # main
controlling function - used to be the main routine
  #library(RSQLite)
  library(RODBC)
  channel<-odbcConnect("wmsdb")
  wmstables<-sqlTables(channel)

  LoGso <- c("river", "dam_lake", "spring", "wetland", "estuary_sea",
             "watersupply", "wastewater", "mine_industry", "agri", "transfer",
             "class_pending", "meteo", "ground")
  if(chem.333) LoGso <- LoGso[LoGso != "ground"]

  # seems to be a mis-reference of ID and RefCode - watch out when linking to WMS via
ODBC:
  if (regiontype=="PDG") s <- s[order(s$Qat, s$ID, s$PointID), ]

  #  Note that WMS Region is the WMS quaternary region - not right
  if (regiontype=="WMA") s <- s[order(s$WMANUM, s$Qat, s$ID, s$PointID), ]

  s$pri <- substr(s$Qat, 1, 1) # Define primary drainage regions from quaternary
  s$sec <- substr(s$Qat, 1, 2) # Define secondary drainage regions from quaternary
  LoTs <- subset(s$Located_On_Type, !duplicated(s$Located_On_Type)) # List of site types
  s <- Located_On_Group(s)
  LoGs <- subset(s$Located_On_Group, !duplicated(s$Located_On_Group)) # List of site
groups
  if (regiontype=="PDG") {
    pris <- s$pri[!duplicated(s$pri)]
  }
  if (regiontype=="WMA") {
    pris <- s$WMANUM[!duplicated(s$WMANUM)]
    pris <- pris[!is.na(pris)]
  }

  for (pr in pris) { # for each primary or management region, create a new KML file,
including a Maucha symbol version
    # for (pr in c("A")) { # test line for limited data set - crashes though:
    # Error in pr + 1 : non-numeric argument to binary operator
    if (regiontype=="PDG") spr <- subset(s, s$pri==pr)
    if (regiontype=="WMA") spr <- subset(s, s$WMANUM==pr)
    KMLhead (pr, spr, regiontype, msym, chem.333, txt.htp, txt.url, txt.hri)
    if(!msym) HTMhead (pr, spr, regiontype, chem.333, txt.htp, txt.url, txt.hri)
    ltys<-subset(spr$Located_On_Type, !duplicated(spr$Located_On_Type)) # List of site
types
    #lgps<-subset(spr$Located_On_Group, !duplicated(spr$Located_On_Group)) # List of site
groups
    for (lgp in LoGso) { # for each group in predetermined sort order
      secs <- 0
      lgps <- subset(spr$Located_On_Group, !duplicated(spr$Located_On_Group)) # List of
site groups
      if(lgp %in% lgps) {
        secs <- subset(spr$sec, !duplicated(spr$sec))
        GRPhead(lgp, pr, msym, regiontype, chem.333)
        for (se in secs) {
          sse <- subset(spr, spr$Located_On_Group==lgp & spr$sec==se)
          #sse<-sse[order(sse$RefCode,sse$PointID,na.last=TRUE),] # put those with a
hydro number first
          # put those with a hydro number first: (no use, messes up catchment order)
          sse <- sse[order(sse$sec, sse$ID, sse$PointID, na.last=TRUE),]
          if(nrow(sse) > 0) {
            SEChead(pr, lgp, se, sse, msym, chem.333)
            # for each item, create an entry in the KML file and HTML file
            SECbody(pr, lgp, sse, channel, msym, chem.333)
            SECtail(pr, se, lgp, msym, chem.333)
          }
        }
        GRPtail(lgp, pr , msym, chem.333)
      }
    }
    KMLtail(pr, msym, chem.333)
    if(!msym) HTMtail(pr, chem.333)
  }
}
```

The main controlling function accesses the Water Management System (WMS) database using open database connectivity (ODBC). The WMS has about 40 types of monitoring site, and in order to simplify the map legend, these are grouped into 12 categories (Table 2).

Table 2. Grouping of Located_On_Type into classes.

Group	Classes of Located_On_Type
class_pending	Class pending, Formal, unknown
meteo	Meteorology
ground	Borehole, Other Ground Fractures, Sinkhole, Dug Well, Well Point, Excavation - Quarry
transfer	Pipeline, Canal, Pump Station, Unknown Transfer Feature Type, Storm Water System, Tunnel, Lateral Collector
watersupply	Potable Water Treatment Works
mine_industry	Mine Property, Mineral Process Plant/Area, Industrial Property, Slime/Slurry Dam, Evaporation Dam, Containment/Emergency/Return Water Dam, Mine Shaft, Intensive Livestock/Irrigated Cropping
river	Rivers
spring	Spring/Eye
dam_lake	Dam / Barrage, Reservoir, Lake
estuary_sea	Estuary/Lagoon, Ocean / Bay
wastewater	Waste Water Treatment Works, Solid Waste Transfer Site, Water/Effluent Treatment Plant, Oxidation Pond
wetland	Pan, Wetland, Nature Site

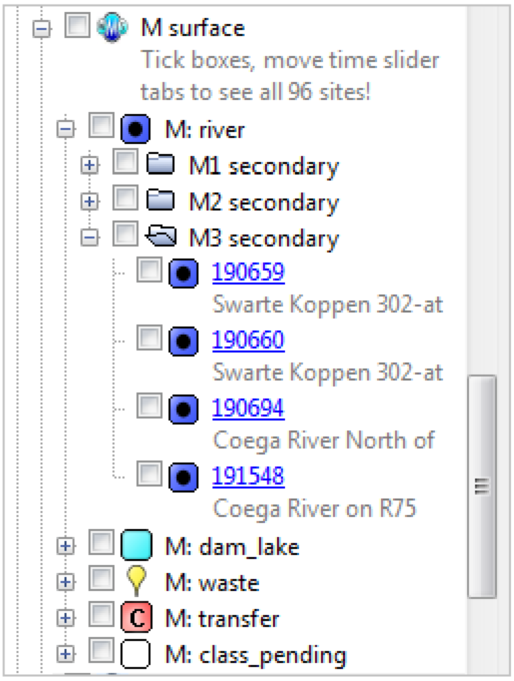

Figure 1. The hierarchical table of contents in Google Earth.

The controlling function processes each region in turn, i.e. primary drainage regions A to X and water management areas 1 to 19. Groundwater is dealt with separately because of the large number of sites, each with only one or a few records. The first step in the procedure is to call the function that creates the headers of the output files in KML and HTML format. Within each region, the function processes the 12 groups of sites in turn, further subdividing them by secondary drainage region in order to create a navigable hierarchy in the Google Earth table of contents (Figure 1). Within each secondary drainage region, the controlling function calls the SECbody function to locate, format and write the information for each site to the KML and HTML files.

Creating KML files: KMLhead, GRPhead, SEChead, SECbody, SECtail, GRP tail

Keyhole Markup Language, KML, is an extensible markup language (XML) for presenting spatial data with an optional time component. The coordinate system allows for portrayal of objects on or above the Earth's surface (Google 2018).

KMLhead

```
KMLhead <- function(pr, spr, regiontype, msym, chem.333, txt.htp, txt.url, txt.hri) { # write the header of the extensible markup language (XML) keyhole
  icons <- ico()
  dwa <- "Department of Water and Sanitation"
  WMA_on <- FALSE # switch for generating WMA files - clumsy but it works
  if (regiontype == "WMA") WMA_on <- TRUE
  if (WMA_on) reg <- wmaname()[pr+1] else reg <- pr
  print(paste("***", format(Sys.time(), "%Y-%m-%d %H:%M:%S"), "KML ",reg, "region", nrow(spr), "monitoring sites", mtxt(msym) , "***"))
  for(lgp in c("ground","surface")) {
    f <-fn (lgp,"KML", pr, msym, chem.333)
    if(lgp == "ground") rows <- nrow(subset(spr, spr$Located_On_Group == "ground"))
    else rows <- nrow(subset(spr, spr$Located_On_Group != "ground"))
    write ("<?xml version=\"1.0\" encoding=\"UTF-8\"?>", file=f)
    write ("<kml xmlns=\"http://earth.google.com/kml/2.2\"\n  xmlns:atom=\"http://www.w3.org/2005/Atom\">", file=f, append=TRUE)
    write (paste("<Document>\n<atom:author>\n  <atom:name>", dwa, "</atom:name>\n</atom:author>", sep=""), file=f, append=TRUE)
    write (paste0("\n<atom:link href=\"", txt.htp, txt.url, txt.hri, "wms/data/000key.asp\" />\n"), file=f, append=TRUE)
    if(WMA_on) write (paste("<name>", wmaname()[pr+1], lgp, "water", dwa, "</name>"), file=f, append=TRUE)
    else write (paste("<name>", pr, lgp, "water", dwa, "</name>"), file=f, append=TRUE)
    write (paste("<Snippet maxLines=\"2\">Tick boxes, move time slider tabs to see all", rows, "sites!</Snippet>"), file=f, append=TRUE)
    write ("<Style id=\"wmsPlacemark\">", file=f, append=TRUE)
    write ("<ListStyle>", file=f, append=TRUE)
    write ("<ItemIcon>", file=f, append=TRUE)
    write (paste0("    <href>", txt.htp, txt.url, txt.hri, "wms/data/wms_m.jpg</href>"), file=f, append=TRUE)
    write ("</ItemIcon>", file=f, append=TRUE)
    write ("</ListStyle>", file=f, append=TRUE)
    write ("</Style>", file=f, append=TRUE)
    write ("<styleUrl>#wmsPlacemark</styleUrl>", file=f, append=TRUE)
    if(msym) mkey(f, txt.htp, txt.url, txt.hri)
    for (ic in 1:nrow(icons))  {
      write (paste("<Style id=\"", icons[ic,1], "\">", sep=""), file=f, append=TRUE)
      write ("  <IconStyle><scale>0.4</scale>", file=f, append=TRUE)
      write ("    <Icon>", file=f, append=TRUE)
      write (paste("      <href>", icons[ic,2], "</href>", sep=""), file=f, append=TRUE)
      write ("    </Icon>", file=f, append=TRUE)
      write ("  </IconStyle>", file=f, append=TRUE)
      write ("  <BalloonStyle>", file=f, append=TRUE)
      write ("  <bgColor>ccffff</bgColor>", file=f, append=TRUE)
      write ("    <text><![CDATA[", file=f, append=TRUE)
      write ("    <font  face=\"Arial\" color=\"#666666\" size=\"+3\"><b>$[name]</b></font><br />",
             file=f, append=TRUE)
      write ("    <font  face=\"Verdana\">$[description]</font><br />", file=f, append=TRUE)
      #write ("    <small>(Links may be slow to open or offline.)<br />", file=f, append=TRUE)
      write ("    $[geDirections]</small>", file=f, append=TRUE)
      write ("    ]]></text>", file=f, append=TRUE)
      write ("  </BalloonStyle>", file=f, append=TRUE)
      write ("</Style>", file=f, append=TRUE)
    }
    for (ic in 1:nrow(icons)) {
      write (paste("<Style id=\"", icons[ic,1], "L\">", sep=""), file=f, append=TRUE)
      write ("  <ListStyle>", file=f, append=TRUE)
      write ("    <ItemIcon>", file=f, append=TRUE)
      write (paste("      <href>", icons[ic,2], "</href>", sep=""), file=f, append=TRUE)
      write ("    </ItemIcon>", file=f, append=TRUE)
      write ("  </ListStyle>", file=f, append=TRUE)
      write ("</Style>", file=f, append=TRUE)
    }
    write ("<open>1</open>", file=f, append=TRUE)
  }
}
```

A KML file begins with a standard header that declares the file type and version, and defines styles for placemarks. Function `KMLhead` writes these lines, with loops to set up definitions of the icons used on maps and in the table of contents. The script creates output files line by line using write statements, harking back to the earlier awk script on which it is based (Silberbauer 2009 Ch4).

GRPhead

```
GRPhead<-function(lgp, pr, msym, regiontype, chem.333) { # write the folder opening text for the type of region, i.e. water management area or primary drainage region
  WMA_on <- FALSE # switch for generating WMA files - clumsy
  if (regiontype=="WMA") WMA_on <- TRUE

  f <- fn(lgp, "KML", pr, msym, chem.333)
  write ("<Folder>", file=f, append=TRUE)
  if(WMA_on) write (paste("  <name>WMA", sprintf("%2.2d",pr), ": ", lgp, "</name>", sep=""), file=f, append=TRUE)
  else write (paste("  <name>", pr,": ", lgp, "</name>", sep=""), file=f, append=TRUE)
  write (paste("  <styleUrl>",lgpIcon(lgp,-1),"L</styleUrl>", sep=""), file=f, append=TRUE)
  write ("  <open>0</open>", file=f, append=TRUE)
  if(msym||lgp=="ground") write ("  <visibility>0</visibility>", file=f, append=TRUE)
}
```

Within the KML file, `<Folder>` statements define the hierarchy in the table of contents (Figure 1) and the `GRPhead` function writes the appropriate heading for a site group class (Table 2). The folder heading includes the style instruction for displaying the appropriate group icon and `<open>0</open>` combination that specifies that the hierarchy will be in a “closed” or compact form when first displayed. If groundwater sites are to be displayed, the tick boxes are left open so that the user can choose which sites to activate. The groundwater sites are numerous and can overload a computer with insufficient memory.

SEChead

```
SEChead<-function(pr, lgp, se, sse, msym, chem.333) {
  # write the folder opening text for the sub-group, i.e. secondary drainage region
  f <- fn(lgp, "KML", pr, msym, chem.333)
  print (noquote(paste(format(Sys.time(), "%Y-%m-%d %H:%M:%S"), f, pr, "[", lgp, "]" , se, nrow(sse) )))
  write ("<Folder>", file=f, append=TRUE)
  write (paste("  <name>", toupper(se), " secondary</name>", sep=""), file=f, append=TRUE)
  write ("  <open>0</open>", file=f, append=TRUE)
  if(msym||lgp=="ground") write ("  <visibility>0</visibility>", file=f, append=TRUE)
}
```

The `SEChead` function writes the `<Folder>` statements for a set of sites in a single secondary drainage region. As in the group header, the hierarchy is “closed” initially, and the user can open it up as in Figure 1. Again, the groundwater sites are not “on” by default.

SECbody

In `SECbody`, the processing of the individual sites within a secondary drainage region takes place. For each row in the site list passed from the main controlling function, the `SECbody` function outputs descriptive information and coordinates. `SECbody` checks the WMS for electrical conductivity data for the site and takes the median as an indicator of salinity and, by implication, general water quality.

The function works out a compact tag for labelling each site. If no site ID is present, the function uses the feature code, replacing anything more than 4 sequential zeroes (“`0000`”) with “-” using `gsub`. If the site ID is a hydrological code like `A2H027Q01` (McDonald 1989), the function checks whether it is a river site, a dam site or a treatment works. In the case of a borehole code like `2918AC00056`, the 1:50 000 topographical map identifier is deleted, leaving `56` in this example (Table 3).

Table 3. Examples of map labels generated in SECbody.

Site ID type	Example of full site ID	Truncated map label
Rivers	A2H027Q01	A2H27

Site ID type	Example of full site ID	Truncated map label
Dam	A2R009Q02	A2R9.2
Water treatment works	R3H002S01	R2H2s1
Rivers – no hydro code	1000010300	1-10300
Borehole	2918AC00056	56

Each site entry in the KML file contains an extended label which the user accesses by clicking on the site icon. The pop-up balloon or information box consists of 19 or more items, where applicable for the type of site (Figure 2).

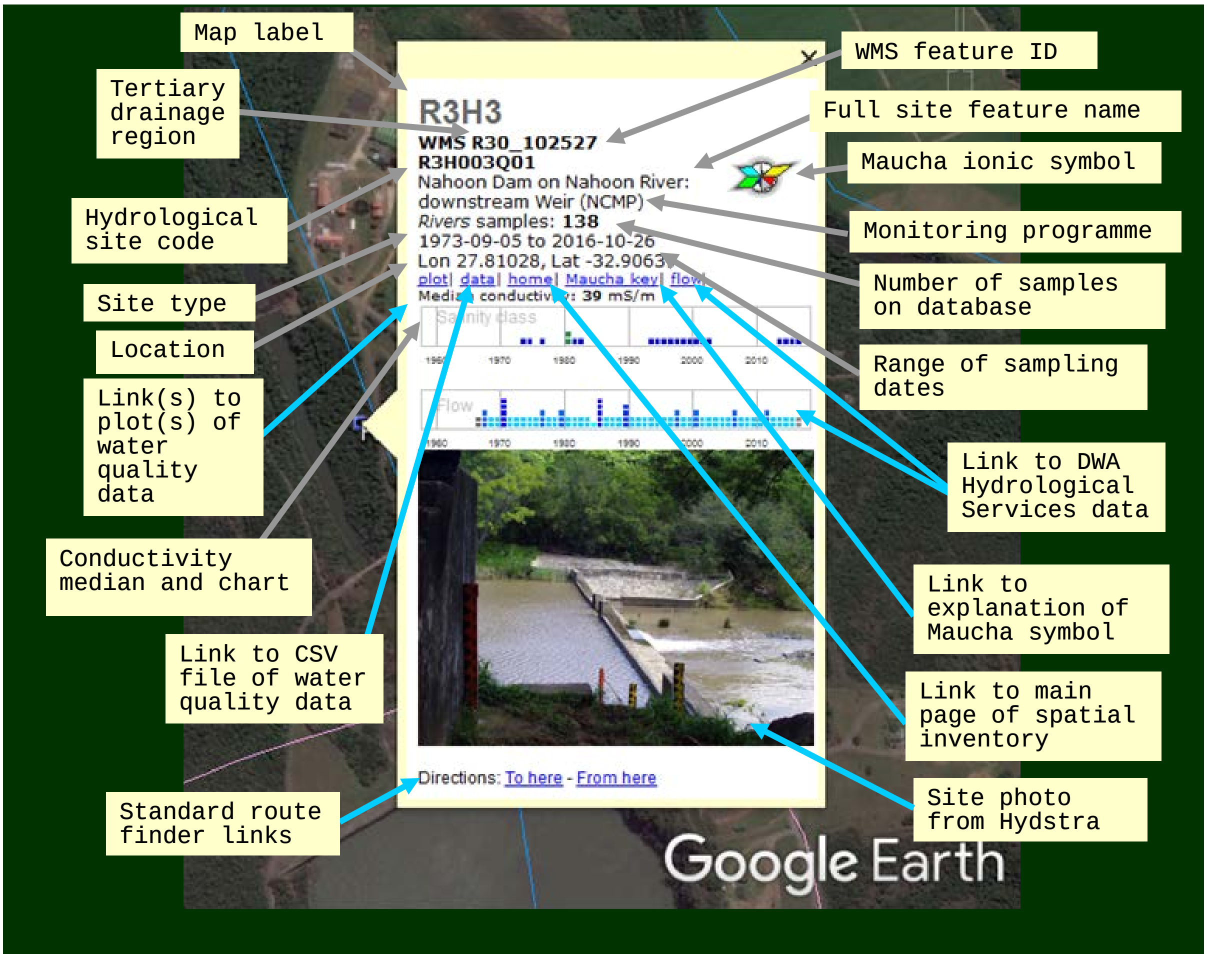

Figure 2. A typical pop-up information box or "balloon".

The following sections explain the `SECbody` code for each item within a pop-up information box. The sequence is that of the information box text, not necessarily that of the R code. The body of the information box text is in HTML format, enclosed in the KML description element and the standard XML CDATA element:

```
<description><![CDATA[ HTML text here ]]></description>
```

Map label

`SECbody` reformats the site label based on its type and writes the name at the start of the pop-up information box:

```
write ("<Placemark>", file=f, append=TRUE)
write (paste("<name>", st, "</name>", sep=""), file=f, append=TRUE)
```

Tertiary drainage region

The tertiary drainage region consists of the first three characters of the quaternary drainage region:

```
Ter <- substr(Qat,1,3)
```

Embedded graphics

This code in `SECbody` builds the filenames of images on the local disk, including Maucha diagrams, and miniplots of electrical conductivity, hydrology and chlorophyll *a*. It also generates the corresponding URL:

```
png.maucha <- paste0("C:/tmp/wms/", Ter, "/", Ter, "_", PointID, "_m.png")
png.ecplot <- paste0("C:/tmp/wms/", Ter, "/", Ter, "_", PointID, "_p.png")
png.fvplot <- paste0("C:/tmp/wms/", Ter, "/", Ter, "_", PointID, "_f.png")
png.chplot <- paste0("C:/tmp/wms/", Ter, "/", Ter, "_", PointID, "_e.png")

chlink <- paste0(txt.htp, txt.url, txt.hri, "eutrophication/NEMP/report/Chart_nemp_",
PointID, ".png")

imglink.maucha <- paste0(txt.htp, txt.url, txt.hri, "wms/data/", Ter, "/", Ter, "_", PointID,
"_m.png")
imglink.ecplot <- paste0(txt.htp, txt.url, txt.hri, "wms/data/", Ter, "/", Ter, "_", PointID,
"_p.png")
imglink.fvplot <- paste0(txt.htp, txt.url, txt.hri, "wms/data/", Ter, "/", Ter, "_", PointID,
"_f.png")
imglink.chplot <- paste0(txt.htp, txt.url, txt.hri, "wms/data/", Ter, "/", Ter, "_", PointID,
"_e.png")
img.hispic <-     paste0(txt.htp, txt.url, "hydrology/Verified/CGI-BIN/HIS/Photos/", sta,
".JPG")
```

The assumption is that any file present on the local disk will have been uploaded to the Internet (Appendix 1 – Uploading to the Internet). The check is as follows:

```
if(file.exists(png.chplot)) {…
```

The following scripts need to run before `kml_WMS_mnpts_html.R`. They prepare embedded graphics and monitoring site shapefiles.

Script	Output
`NCMP_miniplots.R`	compact graphical summaries of electrical conductivity and volume
`Maucha_per_site_chubby.R`	compact summary of ionic ratios
`NEMP_miniplots.R.`	compact graphical summary of summer chlorophyll
`wms2nms.R`	Esri shapefile and Excel spreadsheet of all monitoring sites
`barcode.R`	PNG time series plots of data, CSV files with data, TXT files with metadata

WMS feature_id

`SECbody` outputs the tertiary drainage region and the `feature_id` linked by an underscore:

```
write (paste("<b>WMS ", toupper(Ter), "_", PointID, "</b><br />", sep=""), file=f,
append=TRUE)
```

Site code using numbering convention

If the site has a code of the type used in column two of Table 3, `SECbody` outputs it:

```
write (paste(ref, "<br />", sep=""), file=f, append=TRUE)
```

WMS feature_name

The script writes the site name to the table of contents entry:

```
write (paste("<Snippet maxLines=\"1\">", Name, "</Snippet>", sep=""), file=f,
append=TRUE)
```

and in the information box text:

```
write (paste(Name,"<br />"), sep="", file=f, append=TRUE)
```

Site type and number of samples

`SECbody` combines the site type and number of samples on a single line:

```
write (paste("<i>", Located_On_Type, "</i> samples: <b>", No, "</b><br />",
 sep=""), file=f, append=TRUE)
```

Electrical conductivity

`SECbody` queries the WMS and calculates the median conductivity:

```
vars <- sqlQuery(channel, "select * from monitorng_variable")
    varEC <- subset(vars$mon_variable_id, grepl("EC-Phys-Water",
                    vars$mon_variable_abbr))
    q <- paste("SELECT result_num_value FROM released_result WHERE
                    mon_variable_id = ", varEC,
                   " AND mon_feature_id = ", PointID, sep="")
c <- sqlQuery(channel, q)
    c <- na.omit(c)
if(nrow(c) > 0) medEC <- round(median(c$result_num_value))
    else medEC <- NA
```

If a valid median conductivity value is available, the script outputs the data, colouring the text red if the value is greater than or equal to 350 mS/m:

```
if (!is.na(medEC)) # print (paste(stas, "- no EC data")) else
    {
      if(No == 1)  write ("<br /><small>Conductivity: ", file=f, append=TRUE)
      if(No > 1) write("<br /><small>Median conductivity: ", file=f, append=TRUE)
      if (medEC < 350) { # good to fair water quality
        write (paste("<b>", medEC, "</b> mS/m</small><br />",
                     sep=""),
               file=f, append=TRUE)
      } else {
        write (paste("<b><font color=\"#FF0000\">", medEC,
                     "</font></b> mS/m</small><br />",
                     sep=""),
               file=f, append=TRUE)}
```

Range of sampling dates

The first and last sampling dates come from `sse$FirstDate` and `sse$FirstDate.`

```
if (No == 1) write (paste(FirstDate, "<br />", sep=""), file=f, append=TRUE)
    else write (paste(FirstDate, " to ", LastDate, "<br />", sep=""),
                file=f, append=TRUE)
```

The first and last sampling dates timestamp the point as well:

```
if(No==1) {
      write ("<TimeStamp id=\"ID\">", file=f, append=TRUE)
      write (paste("<when>", FirstDate, "</when>", sep=""), file=f, append=TRUE)
      write ("</TimeStamp>", file=f, append=TRUE)
    }
    else {
      write ("<TimeSpan id=\"ID\">", file=f, append=TRUE)
      write (paste("  <begin>", FirstDate, "</begin>", sep=""), file=f,
append=TRUE)
      write (paste("  <end>", LastDate, "</end>", sep=""), file=f, append=TRUE)
      write ("</TimeSpan>", file=f, append=TRUE)
```

```
    }
```

Coordinates

The site coordinates come from `sse$Latitude` and `sse$Longitude`.

```
sse <- subset(spr, spr$Located_On_Group==lgp & spr$sec==se)
[…]
    Latitude <- sse$Latitude[i]
    Longitude <- sse$Longitude[i]
[…]
write (paste("Lon ", signif(Longitude,7), ", Lat ", signif(Latitude,7), "<br />",
sep=""), file=f, append=TRUE)
```

Link to time-series plot of water quality

A separate script, `barcode.R`, pre-generates a set of time-series plots of inorganic solutes, conductivity and pH from the WMS database (Silberbauer 2018). The files are currently in PNG format for quicker download, because the more crisp PDF files are about ten times larger than the PNG files. `SECbody` checks for the existence of a plot file on the local disk and, if it is available, assumes that the operator has already uploaded it to the Internet, as described under "Embedded graphics".

Link to time-series water quality data

At the same time as it generates the time-series plot for a site, the `barcode.R` script creates a compressed comma-separated value (CSV) file with the data used in the plot, and the metadata. SECbody checks for the existence of the compressed ZIP file on the local disk and, if it is available, assumes that the file is also available on the Internet:

```
zipFile <- paste("C:/tmp/wms/", Ter, "/", Ter, "_", PointID, ".zip", sep="")
zFile <- unlist(strsplit(fn(lgp, "HTM", pr, msym, chem.333), "/"))
zLink <- paste("./", zFile[length(zFile)], sep="")  # take HTML file from path
zipLink <- paste(txt.htp, txt.url, txt.hri, "wms/data/", Ter, "/", Ter, "_",
PointID, ".zip", sep="")if(file.exists(zipFile))
[…]
if(file.exists(zipFile)) write(paste("<a href=\"", zipLink, "\">data</a>|",
sep=""), file=f, append=TRUE)
```

Link to hydrological data

If the site code has the characteristics of a hydrological station as shown in Table 3, `SECbody` generates a link to the Department of Water and Sanitation Hydrology server. Note that from time to time the Hydstra managers may update their site and the link format may change or become inaccessible. This is the code to update if that happens.

```
if(his) {
   fLink <- paste0("\"", txt.htp, txt.url,
"hydrology/Verified/HyDataSets.aspx?Station=", sta, "\"")
   print(paste("Adding link for", sta))
   write(paste0("<a href=", fLink, ">flow</a>|"), file=f, append=TRUE)
}
```

Link to home page of spatial inventory

All pop-up information boxes have a link back to the home page of the inventory site. If any change to the structure of the web site occurs, change the values of txt.htp, txt.url or txt.hri, as needed.

```
write(paste0("<a href=\"", txt.htp, txt.url, txt.hri, "wms/data/000key.htm\"
>home</a>|"), file=f, append=TRUE)
```

Link to explanation of Maucha diagram

If the pop-up information box contains a Maucha diagram, this link takes the user to an explanation of the origins and meaning of the diagram:

```
MauchaLink <- paste0(txt.htp, txt.url, txt.hri, "gis_apps/maucha.pdf")
    if(file.exists(png.maucha)) {
      write(paste("<a href=\"", MauchaLink, "\">Maucha key</a>|", sep=""),
            file=f, append=TRUE)}
```

If the script is generating a KML with Maucha symbols at the monitoring points, the following lines produce the required style code:

```
if(msym && file.exists(png.maucha)) {
   write (paste("<Style id=\"", st, "\"><IconStyle><scale>0.67</scale>", sep=""),
             file=f, append=TRUE)
   write (paste0("  <Icon>", txt.htp, txt.url, txt.hri, "wms/data/", Ter, "/",
                    Ter, "_", PointID, "_m.png</Icon>"),
             file=f, append=TRUE)
   write ("</IconStyle></Style>", file=f, append=TRUE)
   write (paste("<styleUrl>#", st, "</styleUrl>", sep=""),
             file=f, append=TRUE)
 }
 else
   write (paste("<styleUrl>", lgpIcon(lgp, No), "</styleUrl>", sep=""),
             file=f, append=TRUE)
```

KMLhead

`KMLhead` writes the definition of the information box style at the top of the KML file, within each icon style definition:

```
<styleUrl>#wmsPlacemark</styleUrl>
<Style id="normalPlacemark">
  <IconStyle><scale>0.4</scale>
    <Icon>
      <href>http://maps.google.com/mapfiles/kml/paddle/wht-circle-lv.png</href>
    </Icon>
  </IconStyle>
  <BalloonStyle>
  <bgColor>ccffff</bgColor>
    <text><![CDATA[
    <font  face="Arial" color="#666666" size="+3"><b>$[name]</b></font><br />
    <font  face="Verdana">$[description]</font><br />
    $[geDirections]</small>
    ]]></text>
  </BalloonStyle>
</Style>
```

SECtail and GRPtail

`SECtail` closes off the secondary catchment folder in the KML file:

```
SECtail<-function (pr,se,lgp, msym, chem.333) { # secondary catchment tail text
  write(paste("  </Folder><!-- end of",se,"-->"),
   file=fn(lgp,"KML",pr, msym, chem.333), append=TRUE)
}
```

and `GRPtail` closes off the group

```
GRPtail<-function(lgp, pr, msym, chem.333) { # write the folder closing text
  f <- fn(lgp, "KML", pr, msym, chem.333)
  write("</Folder>", file=f, append=TRUE)
}
```

This brings us to the end of the KML creation code. The script also produces a set of HTML tables to permit access to the plots and data without the need for Google Earth.

Creating HTML tables: HTMhead, SECbody, HTMtail

The HTML functions in `kml_WMS_mnpts_html.R` produce standard HTML text and tables with information about the dataset and links to the stations. As with the KML files, the HTML files are divided by primary drainage region or water management area, and surface water or groundwater. The HTML files could, in future, include Maucha diagrams, at the risk of reducing the download speed.

HTMhead

The `HTMhead` function writes the document head block, CSS style code, corporate labelling, standard links and explanatory text (Figure 3).

```
    write ("<!DOCTYPE html PUBLIC \"-//W3C//DTD XHTML 1.0 Transitional//EN\"
\"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd\">", file=f)
    write ("<html>" , file=f, append=TRUE)
    write ("<head>" , file=f, append=TRUE)
    write("<style type=\"text/css\">" , file=f, append=TRUE)
    write("<!-- " , file=f, append=TRUE)
    write("  @import url(\"style_wms_ge.css\"); " , file=f, append=TRUE)
    write(" -->" , file=f, append=TRUE)
    write("</style>" , file=f, append=TRUE)
    write (paste("<title>"), file=f, append=TRUE)
    if(WMA_on) write (paste(wmaname()[pr+1]," WMA ", format(Sys.time(),
      "%Y-%m-%d %H:%M:%S"),
      bfold, sep=" "), file=f, append=TRUE)
    else write (paste(toupper(pr)," region ", format(Sys.time(),
       "%Y-%m-%d %H:%M:%S"),
       bfold, sep=" "), file=f, append=TRUE)
    write (paste(", RQIS WMS water quality data Department of Water and
Sanitation, ",ver(),"</title>") , file=f, append=TRUE)
[…]
```

SECbody

The `SECbody` function described above (page 9) deals mainly with the data for a single point in the KML file but also includes the following code to write out a one-line site entry in the HTML table.

```
if(!msym) { # no sense in creating a separate HTML for the Maucha process, as the HTML already exists
  # Write the point data to the HTML file
  f <- fn(lgp, "HTM", pr, msym, chem.333)
  write ("<tr><td>", file=f, append=TRUE) # start of table row
  write (paste("<b>", toupper(Ter), " ", PointID, "</b></td>", sep="") , file=f, append=TRUE)
  if(file.exists(plotFile)) write (paste("<td><a href=\"", plotLink, "\">plot</a></td>", sep=""), file=f, append=TRUE)
  else write ("<td><i>n/a</i></td>" , file=f, append=TRUE)
  if(file.exists(zipFile)) write(paste("<td><a href=\"", zipLink, "\">data</a></td>", sep=""), file=f, append=TRUE)
  else write ("<td><i>n/a</i></td>" , file=f, append=TRUE)
  write (paste("<td>", Name, " ", dsite, "</td><td>", Located_On_Type, "</td>", sep="") , file=f, append=TRUE)
  write (paste("<td align=\"right\">", No, "</td>", sep=""), file=f, append=TRUE)
  write (paste("<td>", FirstDate, "</td>", sep=""), file=f, append=TRUE)
  write (paste("<td>", LastDate, "</td>", sep=""), file=f, append=TRUE)
  if (is.na(medEC)) write ("<td>n/a</td>", file=f, append=TRUE)
  else write (paste("<td align=\"right\">", medEC, "</td>", sep=""), file=f, append=TRUE)
  if(his) write (paste("<td><a href=", fLink, ">", sta, "</a></td>", sep="") , file=f, append=TRUE)
  else write (paste("<td><small>", st, "</small></td>") , file=f, append=TRUE)
  write ("<td>" , file=f, append=TRUE)
  write(sprintf ("%10.5f",Latitude) , file=f, append=TRUE)
  write ("</td><td>" , file=f, append=TRUE)
  write(sprintf ("%9.5f",Longitude), file=f, append=TRUE)
  write ("</td></tr>" , file=f, append=TRUE)
}
```

HTMtail

`HTMtail` writes a date and time stamp at the bottom of the page, outputs the javascript for Google Analytics page tracking and closes off the HTML file with </body> and </html>.

water & sanitation

Department:
Water and Sanitation
REPUBLIC OF SOUTH AFRICA

Resource water quality data for region R.

WMS water quality sites home

List of 118 sites, other than groundwater, for region R. (Show groundwater list.)

- **plot** Link to pre-generated water quality PNG or PDF time-series plots, which show the median and 90[th] percentile statistics to minimise the effect of outliers.
- **data** Link to compact, comma-delimited water quality data files(missing data = #N/A; for detection limits, obtain the latest "raw" data from RQIS).
- ***n/a*** means that this system could not find inorganic chemical data (other types may exist). (Obtain the latest data from RQIS.)
- Site **descriptions** are verbatim from the database: please report errors.
- **Type** is the database classification.
- **First date**, **last date** and number of samples shown here, while not current, are more up to date than in the pre-generated graphs and data.
- **median** Electrical Conductivity in milliSiemens per metre.
- **Flow data** (where applicable) are presented courtesy of DWS' Hydrological Services.
- Hints:
  - To zoom to a particular point on Google Earth, highlight the **Latitude** and **Longitude** and copy and paste them into Google Earth's search box.
  - The Water Quality Guidelines may help in interpreting water quality data.

Water quality			Description	Type	n	First date	Last date	med EC	Flow, if any	Latitude	Longitude
R10 102500	plot	data	Tyume River at Goumahashe Reserve	Rivers	230	1971-11-08	2016-03-01	20	R1H001	-32.75944	26.85528
R10 102501	plot	data	Keiskamma River at Zanyokwe/Sa Native Trust	Rivers	447	1953-09-14	1995-08-23	21	R1H005	-32.75167	27.09083
R10 102502	plot	data	Keiskamma River at Kammas/Naudeshoek	Rivers	320	1971-11-08	1986-06-03	37	R1H013	-33.01167	26.95472
R10 102503	plot	data	Tyume River at Kwa Khayaletu/Yantolas (NCWQ)	Rivers	865	1971-11-08	2018-03-07	8	R1H014	-32.64000	26.93611
R10 102504	plot	data	Farm 7 About 220M U/S of Howard Shaw Bridge on Keiskamma River (NCWQ NCMP)	Rivers	1128	1971-08-31	2018-03-09	43	R1H015	-33.18536	27.39075
R10 102505	plot	data	Sandile Dam on Keiskamma River: Down Stream Weir (NCWQ)	Rivers	594	1989-07-04	2018-03-07	19	R1H017	-32.71806	27.10639
R10 1000002410	plot	data	Keiskamma River Above St Matthews High School	Rivers	239	2001-08-14	2017-09-13	17	1-2410	-32.64031	27.19058
R10 1000002412	plot	data	Gxulu River Before Confluence with Keiskamma River	Rivers	228	2002-01-29	2017-09-13	21	1-2412	-32.67581	27.14631
R10 1000002413	plot	data	Keiskamma River at R352 Bridge	Rivers	232	2002-03-06	2017-09-13	23	1-2413	-32.68714	27.15239
R10 1000011018	plot	data	Tyume River below Singeni	Rivers	187	2001-11-06	2017-09-13	27	1-11018	-32.87597	26.89256
R10 1000011019	plot	data	Belmont at R72 Howard Shaw Bridge on Keiskamma	Rivers	201	2005-03-09	2017-09-13	30	1-11019	-33.18497	27.39314

Figure 3. Standard table layout. `HTMhead` generates the part up to and including the table header while `SECbody` generates each line within the table.

Special functions

Apart from the main functions in `kml_WMS_mnpts_html.R` dealt with above, the script contains several special service functions (Table 4).

Table 4. Special functions in kml_WMS_mnpts_html.R

Service function	purpose
ver	version numbering and description of changes
fn	compose the required KML and HTML file names based on region and types
ico	set the names and locations of monitoring site marker files
wmaname	set up a list of water management areas in numerical order
Located_On_Group	define groups of Located_On_Type definitions to reduce map legend clutter
lgpIcon	compose the group icon code as defined in ico
mtxt	set the automatic text to display when Maucha diagrams are present
mkey	write the KML code to display a Maucha key at the lower left of the screen
WmsCheck	check for errors in the result of a database query
monitoringSites	read in all monitoring site data (sourced from `monitoringSites.R`)

Results

Conversion of the original `awk` script to R was successful, and the first version was ready in June 2011. Testing took place in July and August 2011 and the R-generated system was live at http://www.dwa.gov.za/iwqs/wms/data/000key.asp by 13 September 2011, six years after the original ArcInfo-awk version went live. Google Analytics registered 465 visits to the inventory from 15 September to 15 November 2011, and no complaints or queries were received.

The KML format is quite versatile and, apart from Google Earth, displays with some success in Google Maps (Figure 5) and ArcGIS.com (Figure 6). It may work in other applications that are able to interpret KML.

Discussion

RODBC opened up opportunities for direct query of the WMS database when assembling data for the KML files. For example, the electrical conductivity entry would have involved too many intermediate steps to be practical in the original `awk` version of the script.

The use of cascading style sheets (CSS) for tabular information produces a neater and more compact display (Figure 4). Although these are nothing new and should have been part of the original system, R provides a more versatile environment for experimenting with and incorporating different methodologies.

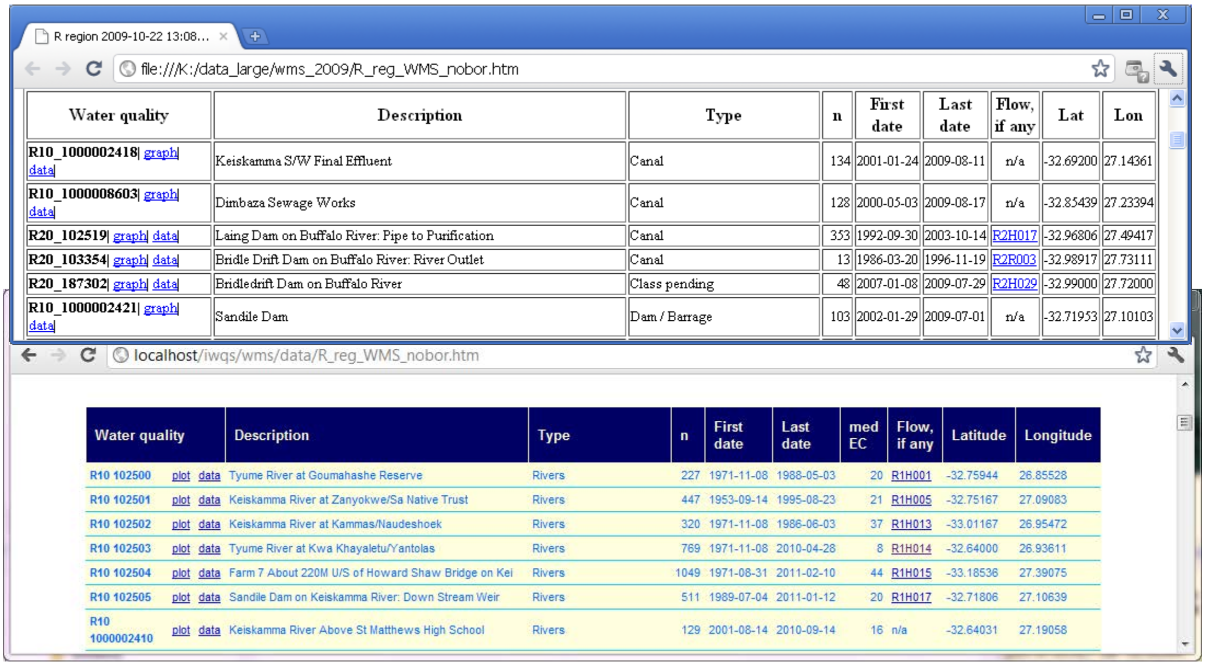

Figure 4. Old table appearance compared with neater format using CSS.

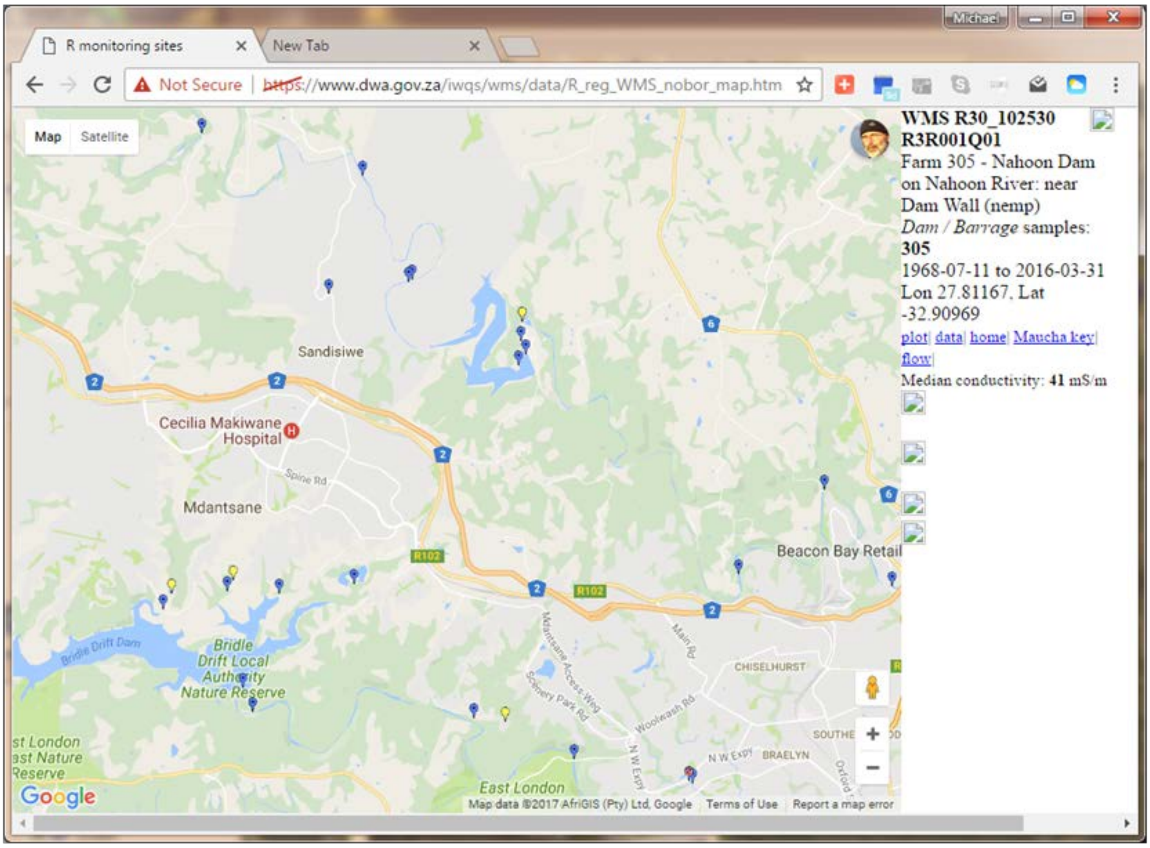

Figure 5. KML file from Figure 2 displayed in Google Maps. This layout is less informative than the previous version of Google Maps (Silberbauer 2011).

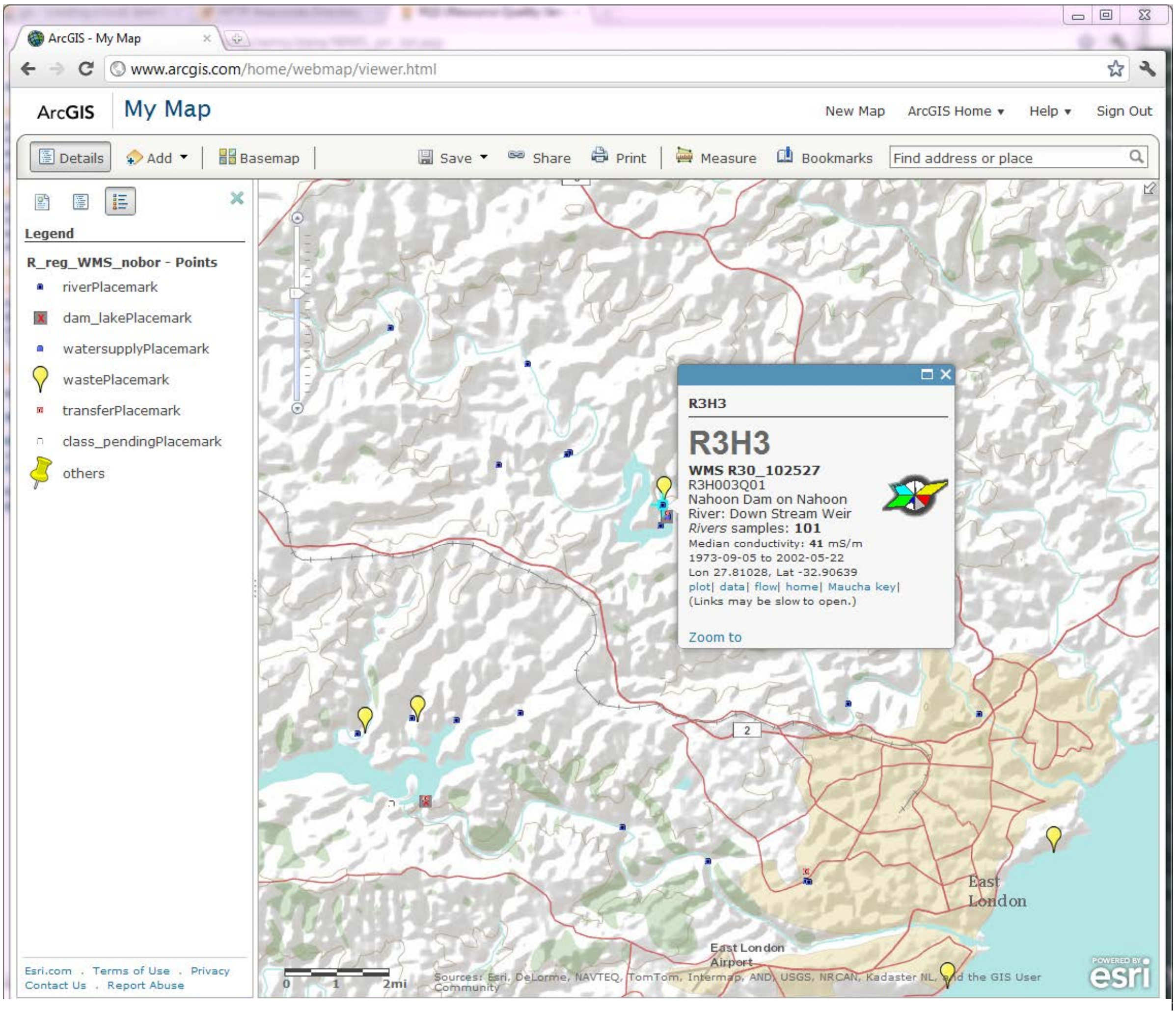

Figure 6. KML file from Figure 2 displayed in ArcGIS.com.

Conclusion

The conversion of kml_WMS_mnpts_html.awk to kml_WMS_mnpts_html.R resulted in more manageable and extensible code. Modifications are now more easily implemented.

This document should serve as a useful reference for anyone wanting to update the spatial inventory or port it to another platform, or who plans to implement a similar system for other data sets.

References

Crampton, J. W. 2008. Keyhole, Google Earth, and 3D Worlds: An Interview with Avi Bar-Zeev. *Cartographica* **43**:85-93.

Google 2018. KML reference for KML Version 2.2. https://developers.google.com/kml/documentation/kmlreference

McDonald, R. D. 1989. New numbers for gauging stations, monitoring points and systems as well as for data and information TR 141 (in Afrikaans). Technical report, Department of Water Affairs, Pretoria.

R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Silberbauer, M. J. 2011. KML-based online spatial inventory of water chemistry data—R version (Conversion to R of the application that creates the online spatial inventory of water chemistry data). First draft. RQS, Pretoria. N/0000/00/RD1/2011

Silberbauer, M. J. 2018. Barcode time-series plots of inorganic water chemistry variables—RODBC version of the application that creates Barcode graphs of the main water chemistry variables. Fourth version. RQIS, Pretoria. N/0000/00/RD2/2011.

Silberbauer, M. J. and Geldenhuys, W. G. 2008. Using Keyhole Markup Language to create a spatial interface to South African water resource data through Google Earth. In Proceedings of the FOSS4G 2008 conference. http://www.foss4g.org/index.php/foss4g/2008/paper/view/74/28.

Silberbauer, M. and Geldenhuys, W. 2009. Google Earth - a spatial interface for SA water resource data. *PositionIT*, (April/May 2009): 42-47.

Appendix 1 – Uploading to the Internet

Uploading of the inventory files to the web server is system-specific. Certain web files need to be in place, for example the main water quality data exploration tool page with its .CSS and .ICO files.
http://www.dwa.gov.za/iwqs/wms/data/000key.asp
<link rel='stylesheet' type='text/css' href='/iwqs/Includes/quickmenu_styles.css' />
<link rel='stylesheet' type='text/css' href='/iwqs/Includes/DWAstylesheet.css' />
<link rel="shortcut icon" href="/iwqs/images/favicon.ico" />

The upload commands are stored in a batch file, `robocopy_kml_ncmp.bat`:

```
rem Q:\ is the mapping to the web server drive

rem compress all related R scripts into a single file
pkzipc -add=update C:\tmp\wms\kml_WMS_mnpts_html_R.zip
c:\data\program\R\kml_WMS_mnpts_html.R c:\data\program\R\barcode.R
c:\data\program\R\NCMP_miniplots.R c:\data\program\R\NEMP_miniplots.R
c:\data\program\R\Maucha_per_site_chubby.R

rem upload zipped script files
robocopy /MAXAGE:20181201 /xo /v /np
/log+:"C:\data\WP\COMPUTER\website\c2qkmlncmp.log" C:\tmp\wms\ Q:\wms\data\
kml_WMS_mnpts_html_R.zip

rem upload zipped shapefile of monitoring sites
robocopy /MAXAGE:20181201 /xo /v /np
/log+:"C:\data\WP\COMPUTER\website\c2qkmlncmp.log" C:\data_large\av\ Q:\gis_data\
nms_wms_geo.zip

rem upload zipped KML files as KMZ
robocopy /MAXAGE:20181201 /xo /v /np
/log+:"C:\data\WP\COMPUTER\website\c2qkmlncmp.log" C:\tmp\wms\ Q:\wms\data\ *.kmz

rem upload the HTML files required for My Maps since 2015
robocopy /MAXAGE:20181201 /xo /v /np
/log+:"C:\data\WP\COMPUTER\website\c2qkmlncmp.log" C:\data\program\awk\
Q:\wms\data\ *_reg_WMS_nobor_map.htm

rem upload the HTML files for each region
robocopy /MAXAGE:20181201 /xo /v /np
/log+:"C:\data\WP\COMPUTER\website\c2qkmlncmp.log" C:\tmp\wms\ Q:\wms\data\ *.htm

rem upload unzipped KML files for Arc Earth
robocopy /MAXAGE:20181201 /xo /v /np
/log+:"C:\data\WP\COMPUTER\website\c2qkmlncmp.log" C:\tmp\wms\ Q:\wms\data\ *.kml

rem check for new chem333 sites to upload
robocopy /MAXAGE:20181201 /xo /v /np
/log+:"C:\data\WP\COMPUTER\website\c2qkmlncmp333.log" C:\tmp\wms333\ Q:\wms\data\
*_reg_WMS_nobor_chem333.kmz

rem copy chem333 files to localhost server
robocopy /MAXAGE:20181201 /xo /v /np
/log+:"C:\data\WP\COMPUTER\website\c2qkmlncmp333.log" C:\tmp\wms333\
R:\iwqs\wms\data\ *_reg_WMS_nobor_chem333.kmz

rem for /f "tokens=2 delims==" %%I in ('wmic os get localdatetime /format:list')
do set datetime=%%I
for /f "tokens=2 delims==" %I in ('wmic os get localdatetime /format:list') do set
datetime=%I

rem popup message when job is complete
msg "%username%" %datetime:~0,8%-%datetime:~8,6% NCMP KML upload completed
```